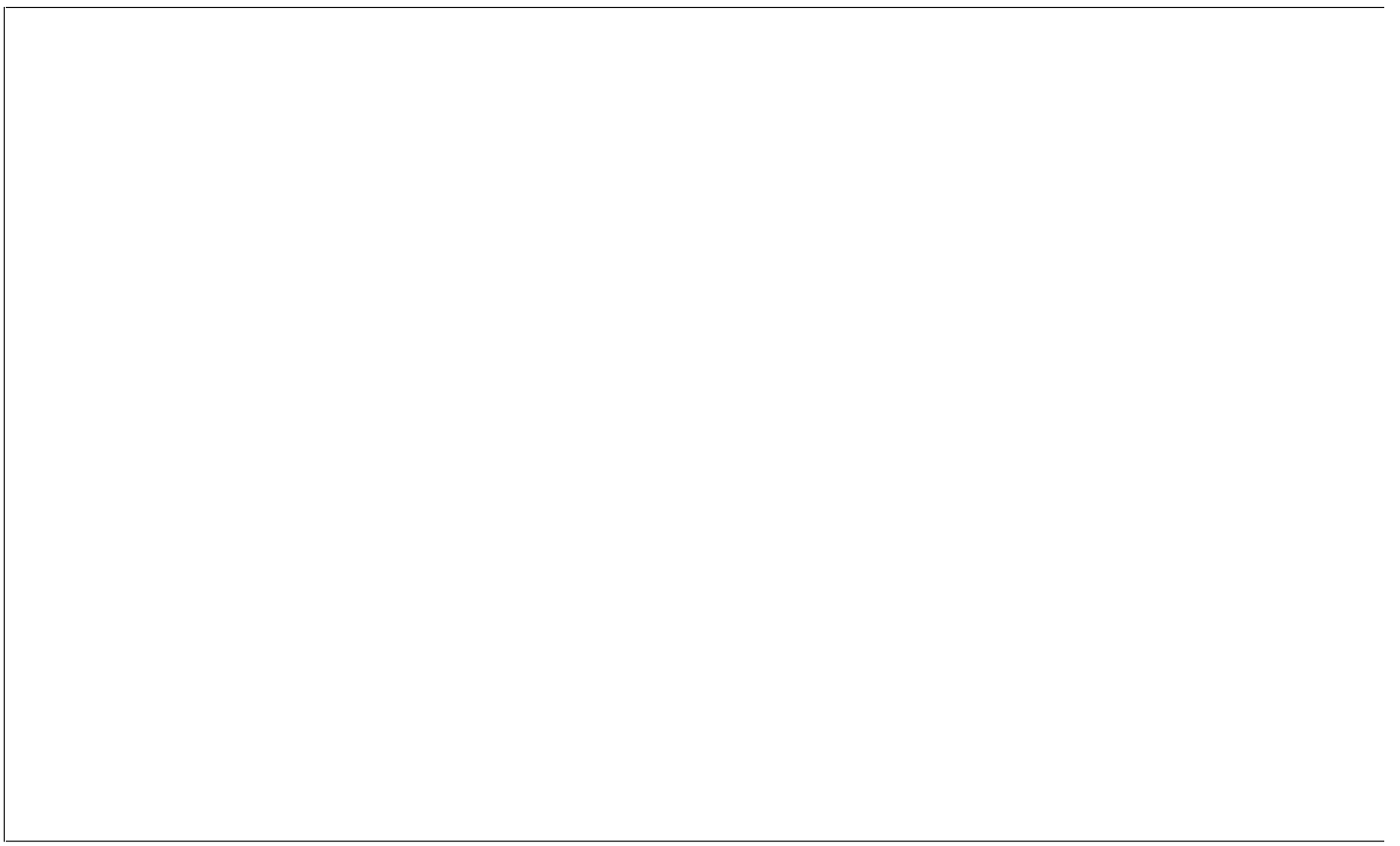
SIEMENS



ICs for Consumer Electronics

TVTEXT 8-Bit Microcontroller, ROMless-Version:
SDA 5250

TVTEXT 8-Bit Microcontroller, ROM-Versions:
SDA 5251
SDA 5252
SDA 5254
SDA 5255

Preliminary Data Sheet 1998-04-08

SDA 525x Revision History:		Current Version: 1998-04-08
Previous Version:		User's Manual 06.97
Page (in previous Version)	Page (in current Version)	Subjects (major changes since last revision)
		The layout of the document has been completely updated.

Edition 1998-04-08

Published by Siemens AG, Bereich Halbleiter, Marketing-Kommunikation, Balanstraße 73, 81541 München

Attention please!

As far as patents or other rights of third parties are concerned, liability is only assumed for components, not for applications, processes and circuits implemented within components or assemblies.

The information describes the type of component and shall not be considered as assured characteristics.

Terms of delivery and rights to change design reserved.

For questions on technology, delivery and prices please contact the Semiconductor Group Offices in Germany or the Siemens Companies and Representatives worldwide (see address list).

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Siemens Office, Semiconductor Group.

Siemens AG is an approved CECC manufacturer.

Packing

Please use the recycling operators known to you. We can also help you – get in touch with your nearest sales office. By agreement we will take packing material back, if it is sorted. You must bear the costs of transport.

For packing material that is returned to us unsorted or which we are not obliged to accept, we shall have to invoice you for any costs incurred.

Components used in life-support devices or systems must be expressly authorized for such purpose!

Critical components[1] of the Semiconductor Group of Siemens AG, may only be used in life-support devices or systems[2] with the express written approval of the Semiconductor Group of Siemens AG.

1 A critical component is a component used in a life-support device or system whose failure can reasonably be expected to cause the failure of that life-support device or system, or to affect its safety or effectiveness of that device or system.

2 Life support devices or systems are intended (a) to be implanted in the human body, or (b) to support and/or maintain and sustain human life. If they fail, it is reasonable to assume that the health of the user may be endangered.

Table of Contents **Page**

Table of Contents **Page**

1 General Description

The SDA 525x contains a slicer for TTX, VPS and WSS, an accelerating acquisition hardware modul, a display generator for “Level 1” TTX data and an 8 bit microcontroller running at 333 ns cycle time. The controller with dedicated hardware guarantees flexibility, does most of the internal processing of TTX acquisition, transfers data to/from the external memory interface and receives/transmits data via I^2C and UART user interfaces. The block diagram shows the internal organization of the SDA 525x. The Slicer combined with dedicated hardware stores TTX data in a VBI buffer of 1 Kbyte. The microcontroller firmware does the total acquisition task (hamming- and parity-checks, page search and evaluation of header control bits) once per field.

2 Features

Acquisition

- Feature selection via special function register
- Simultaneous reception of TTX, VPS and WSS
- Fixed framing code for VPS and TTX
- Acquisition during VBI
- Direct access to VBI RAM buffer
- Acquisition of packets X/26, X/27, 8/30 (firmware)
- Assistance of all relevant checks (firmware)
- 1-bit framing code error tolerance (switchable)

Display

- Features selectable via special function register
- 50/60 Hz display
- Level 1 serial attribute display pages
- Blanking and contrast reduction output
- 8 direct addressable display pages for SDA 5250, SDA 5254 and SDA 5255
- 1 direct addressable display page for SDA 5251 and SDA 5252
- 12×10 character matrix
- 96 character ROM (standard G0 character set)
- 143 national option characters for 11 languages
- 288 characters for X/26 display
- 64 block mosaic graphic characters
- 32 characters for OSD in expanded character ROM + 32 characters inside OSD box
- Conceal/reveal
- Transparent foreground/background - inside/outside of a box
- Contrast reduction inside/outside of a box
- Cursor (colour changes from foreground to background colour)
- Flash (flash rate 1s)

- Programmable horizontal and vertical sync delay
- Full screen background colour in outer screen
- Double size / double width / double height characters

Synchronization

- Display synchronization to sandcastle or Horizontal Sync (HS) and Vertical Sync (VS) with start-stop-oscillator
- Independent clock systems for acquisition, display and controller

Microcontroller

- 8 bit C500-CPU (8051 compatible)
- 18 MHz internal clock
- 0.33 µs instruction cycle
- Parallel 8-bit data and 16...19 - bit address bus (ROMless-Version)
- Eight 16-bit data pointer registers (DPTR)
- Two 16-bit timers
- Watchdog timer
- Capture compare timer for infrared remote control decoding
- Serial interface (UART)
- 256 bytes on-chip RAM
- 8 Kbyte on-chip display-RAM (access via MOVX) for SDA 5250, SDA 5254 and SDA 5255
- 1 Kbyte on-chip display-RAM (access via MOVX) for SDA 5251 and SDA 5252
- 1 Kbyte on-chip TVT/VPS-Acquisition-buffer-RAM (access via MOVX)
- 1 Kbyte on-chip extended-RAM (access via MOVX) for SDA 5250, SDA 5254 and SDA 5255
- 6 channel 8-bit pulse width modulation unit
- 2 channel 14-bit pulse width modulation unit
- 4 multiplexed ADC inputs with 8-bit resolution
- One 8-bit I/O port with open drain output and optional I^2C-Bus emulation (PORT 0)
- Two 8-bit multifunctional I/O ports (PORT 1, PORT 3)
- One 4-bit port working as digital or analog inputs (PORT 2)
- One 2-bit I/O port with optional functions
- One 3-bit I/O port with optional RAM/ROM address expansion up to 512 Kbyte (ROMless-Version)

– **P-SDIP-52-1 Package or P-MQFP-64-1 for ROM-Versions (SDA 5251, SDA 5252, SDA 5254, SDA 5255)**
– **P-MQFP-80-1 Package for ROMless-Version (SDA 5250 M)**
– **P-LCC-84-2 Package for Emulator-Version (SDA 5250)**
– **5 V Supply Voltage**

3 Block Diagram

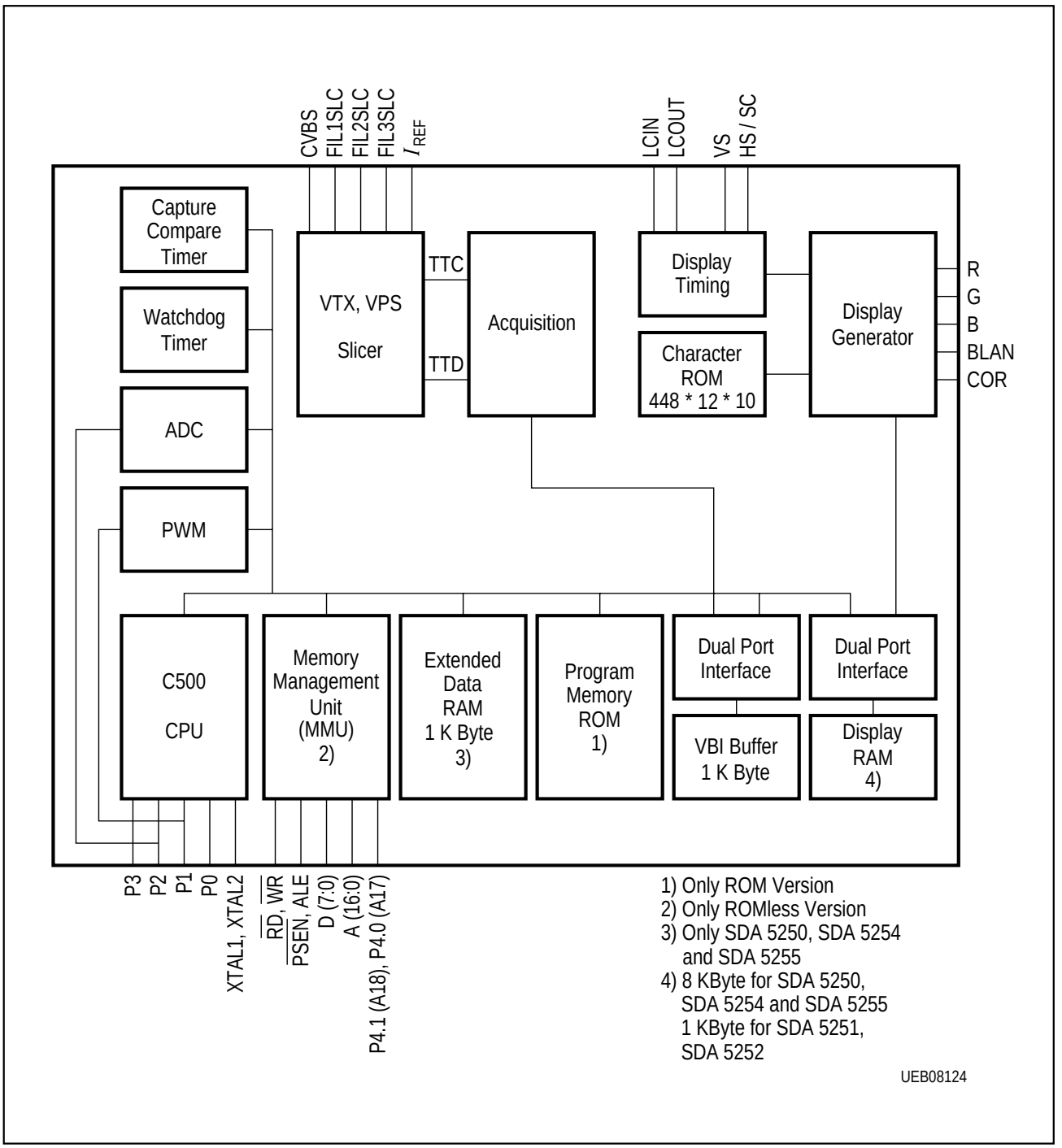


Figure 1
Block Diagram

4 Pin Configurations

4.1 Pin Configuration P-MQFP-80-1 (ROMless-Version)

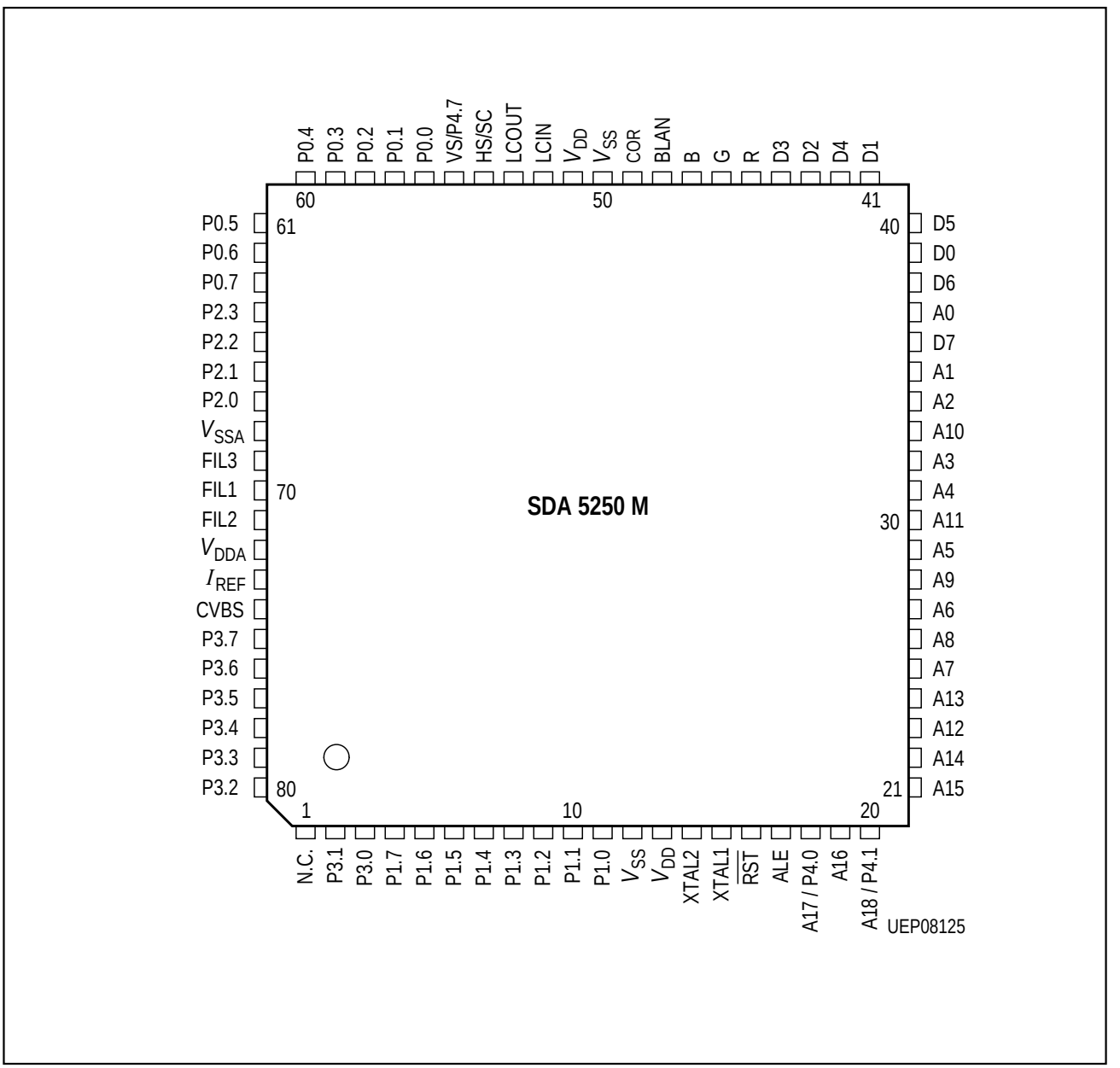


Figure 2
Pin Configuration P-MQFP-80-1 (ROMless-Version)
(top view)

4.2 Pin Configuration P-SDIP-52-1 (ROM-Versions)

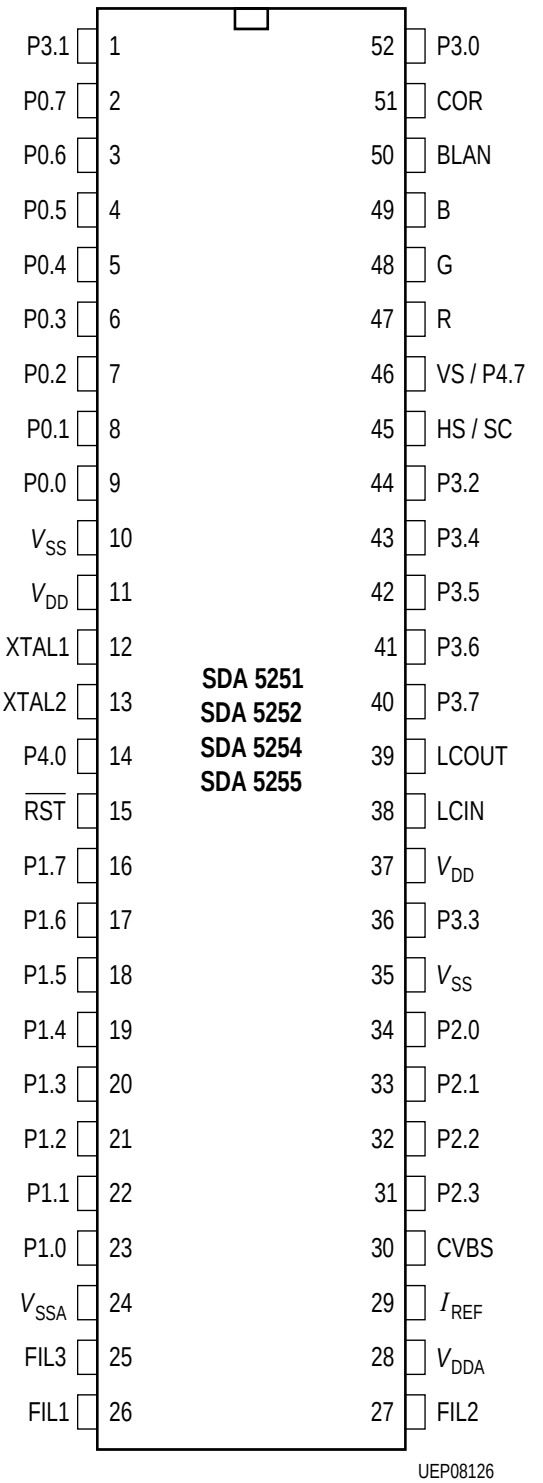


Figure 3
Pin Configuration P-SDIP-52-1 (ROM-Versions)
(top view)

4.3 Pin Configuration P-MQFP-64-1 (ROM-Versions)

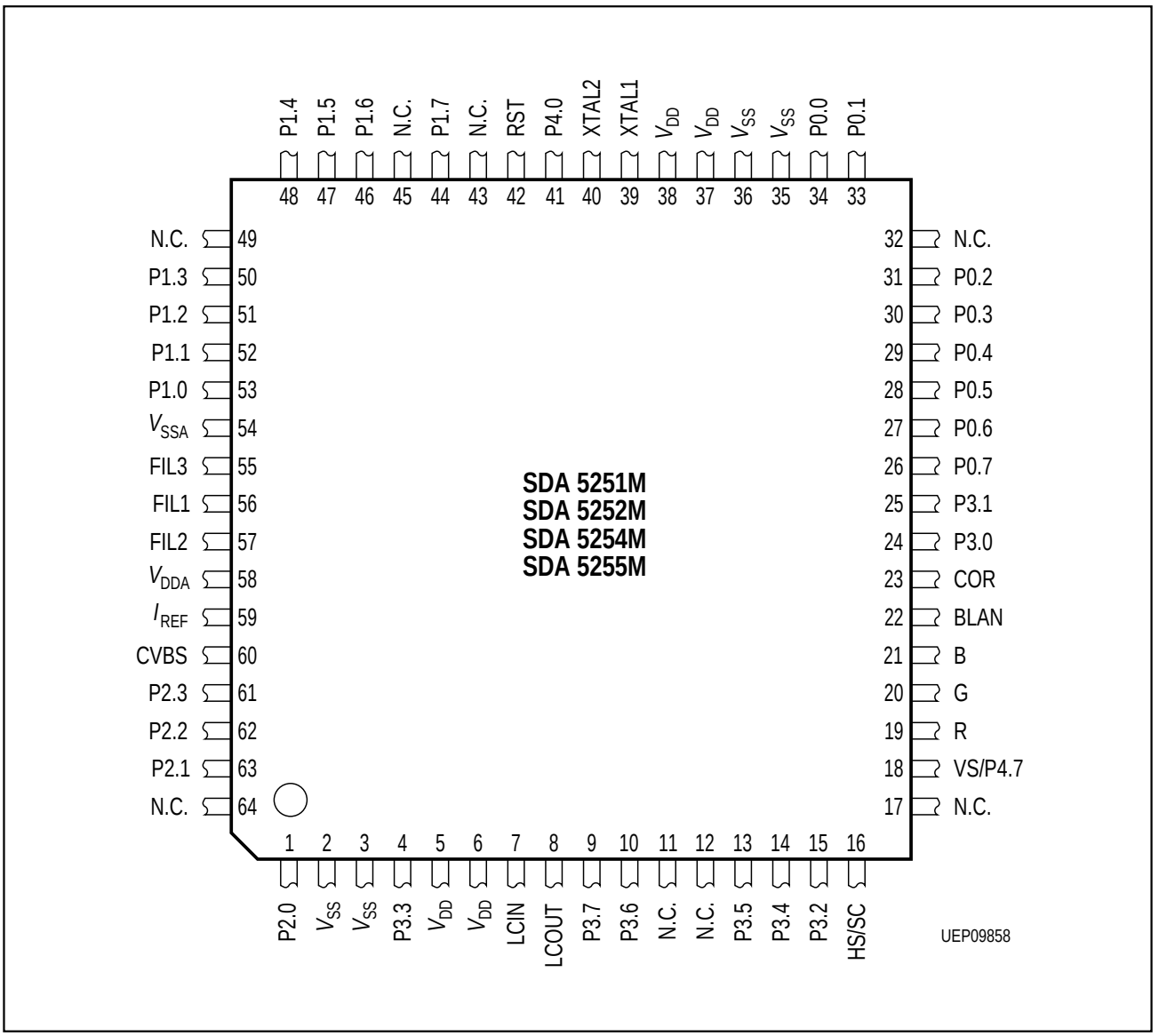


Figure 4
Pin Configuration P-MQFP-64-1 (ROM-Versions)
(top view)

4.4 Pin Configuration P-LCC-84-2 (Emulator-Version)

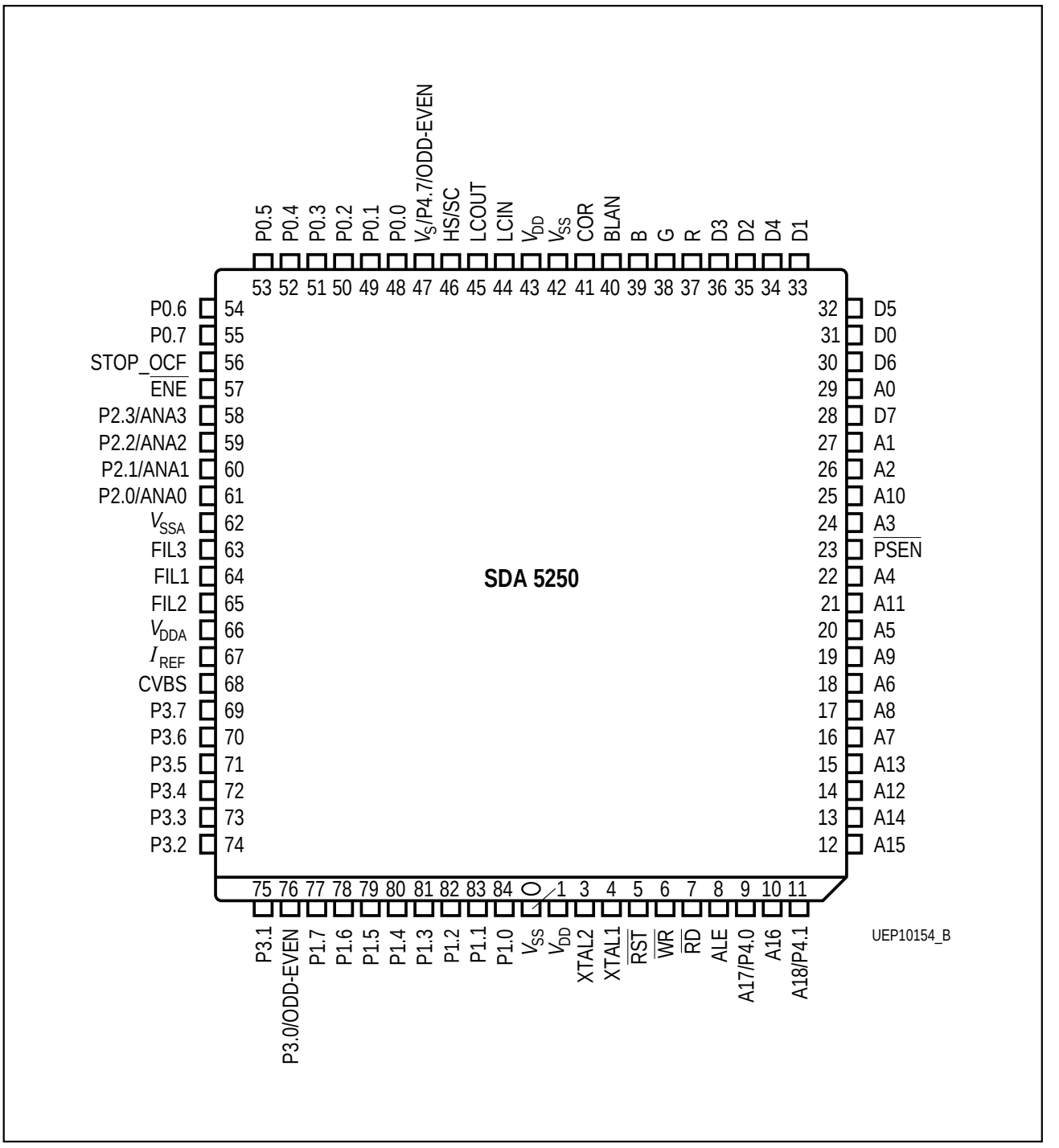


Figure 5
Pin Configuration P-LCC-84-2 (Emulator-Version)
(top view)

5 Pin Functions (ROM- and ROMless-Version)

Table 1
Pin Functions (ROM- and ROMless-Version)

Symbol	Pin No. P-SDIP-52-1	Pin No. P-MQFP-64-1	Pin No. P-MQFP-80-1	Pin No. P-LCC-84-2	Input (I) Output (O) Supply (S)	Function
P0.0	9	34	56	48	I/O	Port 0 is an 8-bit open drain bidirectional I/O port. Port 0 pins that have 1s written to them float; in this state they can be used as high-impedance inputs (e.g. for software driven I^2C Bus).
P0.1	8	33	57	49	I/O	
P0.2	7	31	58	50	I/O	
P0.3	6	30	59	51	I/O	
P0.4	5	29	60	52	I/O	
P0.5	4	28	61	53	I/O	
P0.6	3	27	62	54	I/O	
P0.7	2	26	63	55	I/O	
P1.0	23	53	11	84	I/O	Port 1 is an 8-bit bidirectional multifunctional I/O port with internal pullup resistors. Port 1 pins that have 1s written to them are pulled high by the internal pullup resistors and in that state can be used as inputs. The secondary functions of port 1 pins are: Port bits P1.0 - P1.5 contain the 6 output channels of the 8-bit pulse width modulation unit. Port bits P1.6 - P1.7 contain the two output channels of the 14-bit pulse width modulation unit.
P1.1	22	52	10	83	I/O	
P1.2	21	51	9	82	I/O	
P1.3	20	50	8	81	I/O	
P1.4	19	48	7	80	I/O	
P1.5	18	47	6	79	I/O	
P1.6	17	46	5	78	I/O	
P1.7	16	44	4	77	I/O	
P2.0	34	1	67	61	I	P2.0 - P2.3 are working as digital or analog inputs.
P2.1	33	63	66	60	I	
P2.2	32	62	65	59	I	
P2.3	31	61	64	58	I	
XTAL2	13	40	14	3	O	Output of the inverting oscillator amplifier. To drive the device from an external clock source, XTAL1 should be driven, while XTAL2 is left open.
XTAL1	12	39	15	4	I	Input to the inverting oscillator amplifier
RST	15	42	16	5	I	A low level on this pin resets the processor

Table 1
Pin Functions (ROM- and ROMless-Version) (cont'd)

Symbol	Pin No. P-SDIP-52-1	Pin No. P-MQFP-64-1	Pin No. P-MQFP-80-1	Pin No. P-LCC-84-2	Input (I) Output (O) Supply (S)	Function
V_{DD}	11, 37	5, 6, 37, 38	13, 51	2, 43	S	Power supply voltage
V_{SS}	10, 35	2, 3, 35, 36	12, 50	1, 42	S	Ground (0 V)
R	47	19	45	37	O	Red colour signal output
G	48	20	46	38	O	Green colour signal output
B	49	21	47	39	O	Blue colour signal output
BLAN	50	22	48	40	O	Blanking output
COR	51	23	49	41	O	Contrast Reduction output
P3.0	52	24	3	76	I/O	Port 3 is an 8-bit bidirectional I/O port with internal pullup resistors. Port 3 pins that have 1s written to them are pulled high by the internal pullup resistors and in that state can be used as inputs. It also contains the interrupt, timer and serial port input pins. The output latch corresponding to a secondary function must be programmed to a one (1) for that function to operate. The secondary functions are assigned to the pins of port 3 as follows: - INT0 (P3.2): interrupt 0 input/timer 0 gate control input - $\overline{INT1}$ (P3.3): interrupt 1 input/timer 1 gate control input - T0 (P3.4): counter 0 input - T1 (P3.5): counter 1 input - RXD(P3.6): serial port receive line - TXT(P3.7): serial port transmit line Attention: P3.6 must not be kept to '0' during reset, otherwise a special test mode will be activated.
P3.1	1	25	2	75	I/O	
P3.2	44	15	80	74	I/O	
P3.3	36	4	79	73	I/O	
P3.4	43	14	78	72	I/O	
P3.5	42	13	77	71	I/O	
P3.6	41	10	76	70	I/O	
P3.7	40	9	75	69	I/O	

Table 1
Pin Functions (ROM- and ROMless-Version) (cont'd)

Symbol	Pin No. P-SDIP-52-1	Pin No. P-MQFP-64-1	Pin No. P-MQFP-80-1	Pin No. P-LCC-84-2	Input (I) Output (O) Supply (S)	Function
HS/SC	45	16	54	46	I	Horizontal sync input (alternative sandcastle sync input) for display
VS/P4.7	46	18	55	47	I/O	Vertical sync input for display (alternative Port 4.7)
CVBS	30	60	74	68	I	CVBS (video signal) input
P4.0	14	41	18	9	I/O	Port 4.0 is a bidirectional I/O port
P4.1	–	–	20	11	I/O	with internal pullup resistors. Port 4 pins that have 1s written to them are pulled high by the internal pullup resistors and in that state can be used as inputs. Attention: P4.0 must not be kept to '0' during reset, otherwise a special test mode will be activated.
I_{REF}	29	59	73	67	I	Reference current input for slicer PLLS
V_{DDA}	28	58	72	66	S	Analog Supply Voltage for Slicer and ADC
V_{SSA}	24	54	68	62	S	Analog Ground for Slicer and ADC
FIL1	26	56	70	64	I/O	PLL loop filter I/O for TTX slicing
FIL2	27	57	71	65	I/O	PLL loop filter I/O for VPS/WSS slicing
FIL3	25	55	69	63	I/O	PLL loop filter I/O for TTX/VPS/WSS data slicing
LCIN	38	7	52	44	I	LCIN and LCOUT are used to
LC-OUT	39	8	53	45	O	connect the external display dot clock frequency reference.

Table 2
Additional PINS for ROMless-Version

Symbol	Pin Nr. P-MQFP-80-1	Pin Nr. P-LCC-84-2	Input (I) Output (O) Supply (S)	Function
A0	37	29	O	Address bus for external memory
A1	35	27	O	
A2	34	26	O	
A3	32	24	O	
A4	31	22	O	
A5	29	20	O	
A6	27	18	O	
A7	25	16	O	
A8	26	17	O	
A9	28	19	O	
A10	33	25	O	
A11	30	21	O	
A12	23	14	O	
A13	24	15	O	
A14	22	13	O	
A15	21	12	O	
A16	19	10	O	
D0	39	31	I/O	Data bus for external memory
D1	41	33	I/O	
D2	43	35	I/O	
D3	44	36	I/O	
D4	42	34	I/O	
D5	40	32	I/O	
D6	38	30	I/O	
D7	36	28	I/O	
STOP_OCF	–	56	I/O	Control Signals for data memory extension and emulation.
ENE	–	57	I	
RD	–	7	O	
WR	–	6	O	
ALE	17	8	O	
PSEN	–	23	O	

6 Functional Description

6.1 Acquisition

6.1.1 TTX/VPS Slicer

The slicer extracts horizontal and vertical sync information and TTX data from the CVBS signal. The slicer includes an analog circuit for sync filtering and data slicing. Further there are two analog PLLs for system clock generation for both TTX and VPS. Therefore the slicer is able to receive both TTX and VPS in succeeding lines of a vertical blanking interval. A third data-PLL shifts the phase of the system clock for data sampling. The internal slicer timing signals are generated from the VPS-PLL.

6.1.2 Acquisition Hardware

The acquisition hardware transforms the sliced bit stream into a byte stream. A framing code check follows to identify a TTX or VPS line. If the framing code error tolerance is enabled then one-bit errors will be allowed.

For each line in the VBI in which a framingcode is detected, a maximum of 42 bytes (VPS: 26 bytes) plus a status word are stored in the VBI-buffer. After framing code detection a status word is generated which informs about the type of data received (TTX or VPS) and the signal quality of the TV channel. **Chapter “Acquisition Status Word” on page 17** shows the format of this status word. The horizontal and vertical windows in which TTX or VPS data are accepted and checked for framing code errors are generated automatically. The VBI buffer data will be analyzed (Hamming, parity and acquisition) by the microcontroller and stored in the dual port display RAM or the external RAM, if selected. This analysis is repeated for every field.

Acquisition Status Word

TTX/VPS	FCER	FCOK	LIN.4	LIN.3	LIN.2	LIN.1	LIN.0

LIN.(4...0) number of TV line in which data was received. This information can be used to realize a “software data entry window”. $6 \leq$ LIN.(4...0) ≤ 22

FCOK
1 = Framing code OK (VPS or TTX). This bit is set always by hardware, because lines with valid framing codes are stored only. This bit is reset by software in VBI-buffer. If reset, it indicates that this line was already processed.

FCER
1 = The framing code for TTX lines was accepted with 1-bit-error. For VPS lines this bit has no meaning.
0 = For TTX lines the framing code $E4_H$ was detected.

TTX/VPS
1 = A valid TTX framing code was detected and the data-PLL is locked to the TTX frequency.
0 = A valid VPS framing code was detected and the data-PLL is locked to the VPS frequency.

6.1.3 Memory Interface

The acquisition dual port interface manages the VBI memory write access request from the acquisition hardware and an asynchronous memory access request from the microcontroller. The acquisition hardware delivers the address and data and then a request to the interface. The access of acquisition hardware and controller is under a special arbiter control. The end of data is indicated by the bit LIN24ST in SFR ACQSIR.

6.1.4 Acquisition Control Registers

The following sections gives an overview about special function registers ACQMS_1, ACQMS_2 and ACQSIR, with which slicer and acquisition can be controlled:

Acquisition Mode and Status Register ACQMS_1

Acquisition Mode and Status Register **ACQMS_1** **SFR-Address $C1_H$**

Default after reset: 00_H

(MSB) (LSB)

0	**0**	**VPSE**	**0**	**CRIC.1**	**CRIC.0**	**ENERT**	**TTXE**

TTXE 1: enable TTX in lines 6 - 22

ENERT 1: allow 1 bit error for TTX

CRIC.1 ... CRIC.0
00: The CRI is not included in FRC
01: last 2 bits of CRI are included in the FRC
10: last 4 bits of CRI are included in the FRC
11: last 8 bits of CRI are included in the FRC

VPSE 1: enable VPS in line 16. Text-reception in this line is automatically disabled

Comments Bits 4, 6 and 7 are not defined, must be set to 0

Acquisition Mode and Status Register ACQMS_2

Acquisition Mode and Status Register **ACQMS_2** **SFR-Address $C2_H$**

Default after reset: 00_H

(MSB) (LSB)

TEST.7	**TEST.6**	**TEST.5**	**TEST.4**	**TEST.3**	**TEST.2**	**TEST.1**	**TEST.0**

Comments all bits have to be set to 0. Setting any of these bits will switch on special slicer test modes for production test

Acquisition-Sync-Interrupt-Register ACQSIR

Acquisition-Sync-Interrupt-Register **ACQSIR** **SFR-Address $C0_H$**

Default after reset: 00_H

(MSB) (LSB)

EVENEN	EVENST	LIN24EN	LIN24ST	AVIREN	AVIRST	AHIREN	AHIRST

AHIRST 1 = acquisition horizontal sync interrupt request. This bit is set by the positive edge of HS. It must be reset by software.

AHIREN 1 = enable acquisition horizontal sync interrupt request.

AVIRST 1 = acquisition vertical sync interrupt request. This bit is set by the positive edge of VS. It must be reset by software.

AVIREN 1 = enable acquisition vertical sync interrupt request.

LIN24ST 1 = acquisition line 24 interrupt request. Acquisition hardware processing in VBI interval is finished. This bit assists the synchronization of acquisition software to the ACQ-Timing. It is set by hardware at the beginning of line 24 and the corresponding line of 2nd field. It is reset by software.

LIN24EN 1 = enable acquisition line 24 interrupt request.

EVENST 1 = even field interrupt. Must be reset by software.

EVENEN 1 = enable even field interrupt requests.

Comments None

6.2 Display Generator

The display features of SDA525x are similar to the Siemens SDA5248 TTX controller. The display generator reads character addresses and control characters from the display memory, selects the pixel information from the character ROM and translates it into RGB values corresponding to the World Standard Teletext Norm. The national option character bits for 11 languages inclusive X/26 characters are also supported.

6.2.1 Display Format and Timing

A page consists of 25 rows of 40 characters each. One character covers a matrix of 12 horizontal and 10 vertical pixels. The pixel frequency should be 12 MHz corresponding to 1 µs for one character and 40 µs for one row. A total of 250 TV lines are used for TTX display. The display can be shifted horizontally from 0 µs to 21.33 µs with respect to HS and vertically from line 1 (314) to line 64 (377) with respect to VS. The display position is determined by the registers DHD and DVD.

Note: To avoid interferences between the subharmonics of the 18 MHz controller clock and the 12 MHz pixel clock, a pixel clock of about 11,5 MHz is recommended.

6.2.2 Display Cursor

A cursor is available which changes foreground to background colour for one character. Cursor flash can be realized via software enabling/disabling the cursor. The cursor position is defined by cursor position registers DCRP and DCCP.

6.2.3 Flash

A character background flash (character is changed to background colour) is realized by hardware. The flash frequency is 1 Hz with a duty cycle of 32:18.

6.2.4 Full Screen Background Colour

The SDA 525x delivers the new full screen background colour feature. Special function register SFR DTIM(7-5) includes three bits which define the default background colour for the inner and outer screen area.

6.2.5 Clear Page Logic

The clear page logic generates a signal which is interpreted by the character generator to identify non displayable rows. In row 25 specific information is stored by the microcontroller indicating which of the rows 0 - 24 should be interpreted as erased during character generation. At the beginning of each row the special control characters are read from the display memory (see **Table 3**).

Table 3
Clear Page Bits

row 25 /column:	D7	D6	D5	D4	D3	D2	D1	D0
0	ER_7	ER_6	ER_5	ER_4	ER_3	ER_2	ER_1	ER_0
1	ER_{15}	ER_{14}	ER_{13}	ER_{12}	ER_{11}	ER_{10}	ER_9	ER_8
2	ER_{23}	ER_{22}	ER_{21}	ER_{20}	ER_{19}	ER_{18}	ER_{17}	ER_{16}
3	0	0	0	0	0	0	0	ER_{24}

$ER_{24}...ER_0 = 1$: row is interpreted as a blanked row

$ER_{24}...ER_0 = 0$: row is received and displayed

6.2.6 Display Page Addressing

The display generator hardware generates a row/column address for the display memory. Because there is a binary to row/column address translation between display generator and memory, the OSD programmer has to take care of this. The relationship between row/column and binary address in memory is shown in **Table 4**.

Table 4
Row/Column to Binary Translation Table

	C0	...	C31	C32	...	C39
Row0	00_H	...	$1F_H$	$3F8_H$	...	$3FF_H$
Row1	20_H	...	$3F_H$	$3F0_H$	...	$3F7_H$
:	:	:	:	:	:	:
Row23	$2E0_H$	...	$2FF_H$	340_H	...	347_H
Row24	300_H	...	$31F_H$	338_H	...	$33F_H$
Row25	320_H	...	337_H			

6.2.7 Character Generator

The character generator includes the character and control code decoder, the RAM interface and the RGB-, BLAN- and COR-signal generator. The display generator reads data from the display RAM and calculates appropriate data which drives the RGB output pins. The pixel clock is generated by a start-stop-oscillator. The synchronization of display and pixel clock is done via external sandcastle or HS and VS signals. For 60 Hz display the number of lines per character can be reduced to 9 or 8. In this case pixel information of line 10 or 9 plus 10 are rejected. With this mode combined with the variable vertical offset it is possible to generate NTSC displays with 25 rows.

Characters with a binary value < 32 are interpreted as control characters. For binary values ≥ 32 a ROM character is selected through the addition of the character address, the language setting in SFR, the europe designation and the graphics control bits delivered from the control bit decoder.

A total of 64 OSD characters and 64 mosaic graphics characters are available. OSD characters with addresses 80...SF_H can be displayed together with 60 lower case characters because there is no memory overlapping with any other characters. OSD characters with addresses 60...$7F_H$ can only be displayed if bit OSD in SFR LANGC is set (see diagrams: Physical Address Space and Vertical Address Space).

Figures 6 - **13** shows the character ROM contents.

The control byte decoder analyses the serial attributes from the display memory and generates control clocks for the RGB logic and the character address decoder. The interpretation of control characters is corresponding to World Standard Teletext norm. **Table 5** shows the characters and the appearance on the screen.

The RGB logic combines data from the character address decoder, control byte decoder and settings from the SFR registers and generates signal R, G, B, BLAN and COR.

6.2.8 On Screen Display (OSD)

A display page in the display memory can also be used for on screen displays. It should be recognized that all serial attributes of a normal text page are also valid for an OSD display. Therefore if double height is selected anywhere in a normal text page, row n and row n-1 (upper row) should be saved and overwritten by OSD data in order to generate a correct display. Switching back to text display is accomplished by rewriting the text data to the page. The same procedure is needed for the “erase row bits” in row 25. By means of enable box bits, transparent control bits and the serial attribute “OSD”, the OSD screen can be controlled fully independent of the normal text page. The serial OSD-bit toggles the screen between normal display and OSD.

Table 5
Serial Control Bytes

B7, B6, B5, B4	0	1
B3, B2, B1, B0		
0	Alpha Black	Mosaic Black
1	Alpha Red	Mosaic Red
2	Alpha Green	Mosaic Green
3	Alpha Yellow	Mosaic Yellow
4	Alpha Blue	Mosaic Blue
5	Alpha Magenta	Mosaic Magenta
6	Alpha Cyan	Mosaic Cyan
7	Alpha White(1)	Mosaic White
8	Flash	Conceal(2)
9	Steady(1,2)	Contiguous Mosaic(1,2)
A	End Box (1,3)	Separated Mosaic(2)
B	Start Box (3)	OSD(5)
C	Normal Height(1,2)	Black Background(2)
D	Double Height	New Background (2)
E	Double Width(4)	Hold Mosaic(2)
F	Double Size(4)	Release Mosaic(1)

(1) Reset state at begin of each row.
(2) Takes effect with control character. Other control characters takes effect in the next character field.
(3) Two identical control characters are transmitted in sequence. The effect begins between the control characters.
(4) Can only be activated if SFR DMOD.0 is set to ‘1’, otherwise no influence.
(5) Toggle; takes effect with next character (on), takes effect with control character (off).

6.2.9 Display Special Function Registers

The display generator includes 9 registers to select the different formats and functions.

Display Horizontal Delay Register DHD

Display Horizontal Delay Register **DHD** **SFR-Address $C3_H$**

Default after reset: 00_H

(MSB) (LSB)

HD.7	**HD.6**	**HD.5**	**HD.4**	**HD.3**	**HD.2**	**HD.1**	**HD.0**

HD.7 ... HD.0 variable negative horizontal display offset relative to positive edge of HS in pixel units.

Comments None

Display Vertical Delay Register DVD

Display Vertical Delay Register **DVD** **SFR-Address $C4_H$**

Default after reset: 00_H

(MSB) (LSB)

–	**–**	**VD.5**	**VD.4**	**VD.3**	**VD.2**	**VD.1**	**VD.0**

VD.5 ... VD.0 variable negative vertical display offset relative to positive edge of VS in HS units.

Comments None

Display Transparent Control Register DTCR

Display Transparent Control Register **DTCR** **SFR-Address $C5_H$**

Default after reset: 00_H

(MSB) (LSB)

CORI	CORO	ICRP	IBP	TRFI	TRFO	TRBI	TRBO

TRBO 1 = Transparent Background Colors outside Box **and** OSD.

TRBI 1 = Transparent Background Colors inside Box **or** OSD.

TRFO 1 = Transparent Foreground Colors outside Box **and** OSD.

TRFI 1 = Transparent Foreground Colors inside Box **or** OSD.

IBP 1 = Invert Blanking Polarity. Blanking is active high.
0 = Blanking is active low.

ICRP 1 = Invert Contrast Reduction Polarity. COR is active high.
0 = COR is active low.

CORO 1 = Contrast Reduction for Background Color outside Box and outside OSD.

CORI 1 = Contrast Reduction for Background Color inside Box or inside OSD.

*Note: Outside of a box means outside of a box opened by control code sequence '0B,0B' **and** outside of an OSD-Box opened by control code '1B'.*
*Inside a box means inside of a box opened by control code sequence '0B,0B' **or** inside an OSD-Box opened by control code '1B'.*

Comments For further Transparent Modes see SFR DCRP.

Display Mode Register DMOD

Display Mode Register **DMOD** **SFR-Address $D6_H$**

Default after reset: XXXX0000B

(MSB) (LSB)

–	–	–	–	0	0	0	DSDW

DSDW if set, displaying Double Size and Double Width characters is enabled
if cleared, control codes $0E_H$ and $0F_H$ have no effect

Bit 1 to 3 have always to be written with ‘0’

Bit 4 to 7 not implemented, to be written with ‘0’

Note: This register is not readable. Thus, do not use read-modify-write operations like ANL, ORL to modify this register.

Display Feature *Double Size* and *Double Width*

Double Size and Double Width are selectable via serial attributes. The control codes are ‘0E’ for Double Width and ‘0F’ for Double Size. Now, there are 4 control codes available, to modify the character size:

Control Code	Name	Effect	Side Effects
0C	Normal Size	No stretching	any activated stretching off
0D	Double Height	Vertical character stretching	horizontal stretching off
0E	Double Width	Horizontal character stretching	vertical stretching off
0F	Double Size	Horizontal and vertical stretching	None

Since Double Width and Double Size control codes should not be interpreted by a pure level 1 text-decoder, this size attributes have to be enabled by setting SFR-bit DSDW.

Double Width and Double Size characters are accomplished by skipping every second character code after setting any of this following attributes where the remaining displayable characters are stretched horizontally and thus conceating the character. Although every second character is hidden, these codes will take effect if they are control characters.

Display Mode Register 1 DMODE1

Display Mode Register 1 **DMODE1** **SFR-Address $C6_H$**

Default after reset: 00_H

(MSB) (LSB)

ST_TOP	ST_DIS	CON	DH.1	DH.0	BD_24	BD_1_23	BD_0

BD_0
1 = Box characters in row 0 are ignored.
0 = Box characters in row 0 are displayed.

BD_1_23
1 = Box characters in row 1 - 23 are ignored.
0 = Box characters in row 1 - 23 are displayed.

BD_24
1 = Box characters in status row are ignored.
0 = Box characters in status row are allowed.

DH.1 ... DH.0
00 = Normal row display.
01 = Rows 0 - 11 are displayed in double height. Status row is displayed in normal height.
10 = Rows 12 - 23 are displayed in double height. Status row is displayed in normal height.
11 = Not defined.

CON
1 = Concealed characters are visible.
0 = Concealed characters are not visible.

ST_DIS
1 = Status row is handled as blanked row.
0 = Status row is displayed.

ST_TOP
1 = Status row is displayed in row 0 of display.
0 = Status row is displayed in row 24 of display.

Comments
Only boxes opened by the control code sequence $0B_H$, $0B_H$ will be influenced, an OSD-Box (opened by control code $1B_H$) will not be affected.

Display Mode Register 2 DMODE2

Display Mode Register 2 **DMODE2** **SFR-Address $C7_H$**

Default after reset: 00_H

(MSB) (LSB)

DTEST.2	DTEST.1	DTEST.0	DCHAP.2	DCHAP.1	DCHAP.0	C10	C7

C7 1 = Header is handled as erased row (Suppress Header).

C10 1 = Rows 1 - 23 are handled as erased rows (Inhibit Display).

DCHAP.2..0 selects one of eight display chapters.

DTEST.0 Not defined, must be set to 0.

DTEST.1 Not defined, must be set to 0.

DTEST.2 Not defined, must be set to 0.

Comments For 1-page-versions (SDA 5251, SDA 5252) the bits DCHAP.2..0 have to be set to ‘0’.

Language Control Register LANGC

Language Control Register **LANGC** **SFR-Address $C9_H$**

Default after reset: 00_H

(MSB) (LSB)

OSD_64	LANGC.6	LANGC.5	LANGC.4	LANGC.3	LANGC.2	LANGC.1	LANGC.0

LANGC.4... LANGC.0 Language selection for text outside of an OSD window.
00000 : German
01010 : English
01011 : Scandinavian
01100 : Italian
01101 : French
01110 : Spanish
11001 : Turkish
11010 : Rumanian
11011 : Hungarian
11100 : Czechish
11101 : Polish
11110 : Serbian
others : Not defined

LANGC.6... LANGC.5
00: West european special characters are addressable.
01: West european special characters are addressable (Turkish).
10: East european special characters are addressable.
11: Not defined.

OSD_64
1: 64 OSD character mode on. If the serial attribute OSD is set a total of 64 OSD characters is available. The lower case G0 characters can not be used.
0: 32 OSD character mode on. Only OSD characters in ROM column 8 and 9 are available if serial attribute OSD is set. Outside an OSD box all 64 OSD characters are available (see **Figure 12**).

Comments see Diagrams ‘x’ and ‘y’ Physical and Vertical address spaces

Display Cursor Column Position Register DCCP

Display Cursor Column Position Register **DCCP** **SFR-Address CA_H**

Default after reset: 00_H

(MSB) (LSB)

–	DC_EN	DCCP.5	DCCP.4	DCCP.3	DCCP.2	DCCP.1	DCCP.0

DC_EN 1 = Display Cursor Enable.
0 = Display Cursor Disable.

DCCP.5...DCCP.0 Active cursor column position.
DCCP.5...0 = 0_D: column 1 on screen.

Bit 7 reserved, should be set to ‘0’.

Comments None

Display Cursor Row Position Register DCRP

Display Cursor Row Position Register **DCRP** **SFR-Address CB_H**

Default after reset: 00_H

(MSB) (LSB)

TRBOS	COROS	–	DCRP.4	DCRP.3	DCRP.2	DCRP.1	DCRP.0

DCRP.4...DCRP.0 defines row of active cursor position.

TRBOS 1 = The outer screen display area appears transparent
0 = The outer screen display area gets the background colour defined in register DTIM

COROS 1 = Contrast reduction outer screen

Bit 5 reserved, should be set to ‘0’.

Comments bits TRBOS and COROS thematically belong to the SFR DTCR

Display Timing Control Register DTIM

Display Timing Control Register **DTIM** **SFR-Address CC_H**

Default after reset: 00_H

(MSB) (LSB)

BG_R	BG_G	BG_B	EO_P30	EO_VS	SANDC	LIN9	LIN8

LIN8 1 = 8 line character mode (higher priority than LIN9).

LIN9 1 = 9 line character mode.

SANDC 1 = horizontal and vertical synchronization accepts sandcastle pulse from pad HS/SC.
0 = horizontal and vertical synchronization accepts HS and VS pulses from pads HS/SC and VS respectively.

EO_VS 1 = The ODD/EVEN-signal evaluated from CVBS is enabled on Pin VS.
0 = ODD/EVEN function is disabled.

EO_P30 1 = The ODD/EVEN-signal evaluated from CVBS is enabled on Pin P3.0.
0 = ODD/EVEN function is disabled.

BG_R
BG_G
BG_B

	BG_R	BG_G	BG_B		BG_R	BG_G	BG_B
black	0	0	0	yellow	1	1	0
red	1	0	0	violet	1	0	1
green	0	1	0	cyan	0	1	1
blue	0	0	1	white	1	1	1

outer screen background colour

Teletext-Sync-Interrupt-Register TTXSIR

Teletext-Sync-Interrupt-Register **TTXSIR** **SFR-Address $C8_H$**

Default after reset: 00_H

(MSB) (LSB)

–	VSY	HSY	PCLK	DVIREN	DVIRST	DHIREN	DHIRST

DHIRST 1 = display horizontal sync interrupt request (set by positive edge of HS, reset by software).

DHIREN 1 = enable display horizontal sync interrupt requests.

DVIRST 1 = display vertical sync interrupt request (set by positive edge of VS, reset by software).

DVIREN 1 = enable display vertical sync interrupt requests.

PCLK Reflects state of internal pixel clock.

HSY Reflects state of HS-signal decoded by SC-decoder

VSY Reflects state of VS-signal decoded by SC-decoder (SANDC=1).
Reflects state of VS-pin (SANDC=0).

Bit 7 reserved, should be set to ‘0’

6.2.10 Sandcastle Decoder

To fit the requirements of various applications the input circuit of the sandcastle decoder is programmable. Both slicing levels (V_{SCH}, V_{SCL2}) which are important for proper SC-decoder function can be varied in a range of about 0.9 V and in addition there is the possibility to increase the implemented hysteresis by 0.3 V typically. Further noise reduction and spike rejection on pin SC is accomplished by using a digital filter following the input circuitry. See **Figure 41** on **page 133** for further information on V_{SCH} and V_{SCL2}.

Sandcastle Control Register SCCON

Sandcastle Control Register SCCON SFR-Address CE_H

Default after reset: 00_H

(MSB) (LSB)

0	**SCCH.2**	**SCCH.1**	**SCCH.0**	**0**	**SCCL.2**	**SCCL.1**	**SCCL.0**

SCCL1...0 00 = set V_{SCL2} to lowest level (1.0 V typ.)
01 = increase V_{SCL2} by 0.3 V (typ.)
10 = increase V_{SCL2} by 0.6 V (typ.)
11 = increase V_{SCL2} by 0.9 V (typ.)

SCL.2 0 = hysteresis V_{SCL2} set to 0.3 V (typ.)
1 = increase hysteresis V_{SCL2} by 0.6 V (typ.)

SCCH1...0 00 = set V_{SCH} to lowest level 3.0 V (typ.)
01 = increase V_{SCH} by 0.3 V (typ.)
10 = increase V_{SCH} by 0.6 V (typ.)
11 = increase V_{SCH} by 0.9 V (typ.)

SCCH.2 0 = hysteresis V_{SCH} set to 0.3 V (typ.)
1 = increase hysteresis V_{SCH} by 0.6 V

Attention Bits 3 and 7 have to be set to ‘0’.

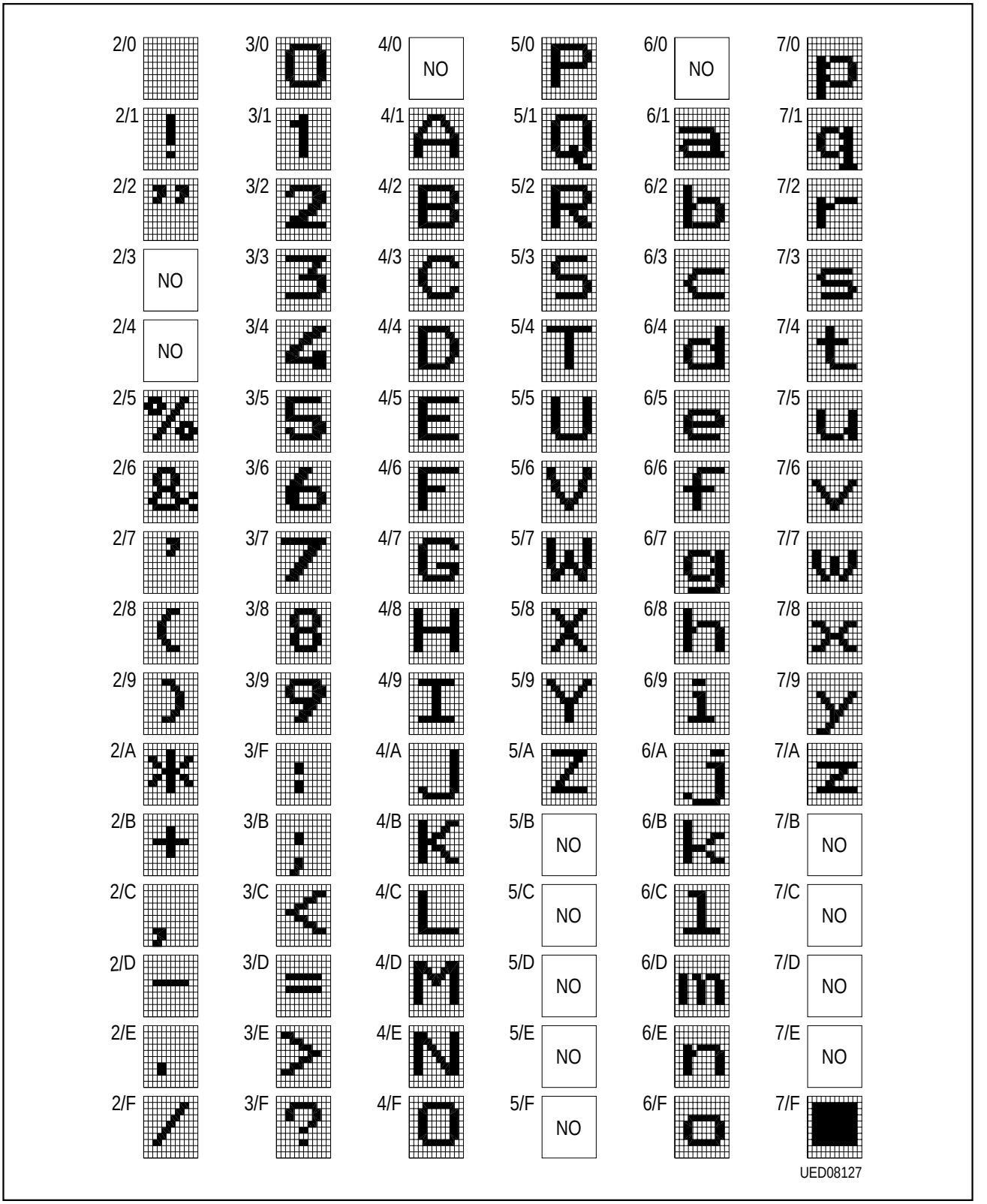


Figure 6
G0 Character Set

Note: NO = hardware mapped national option character

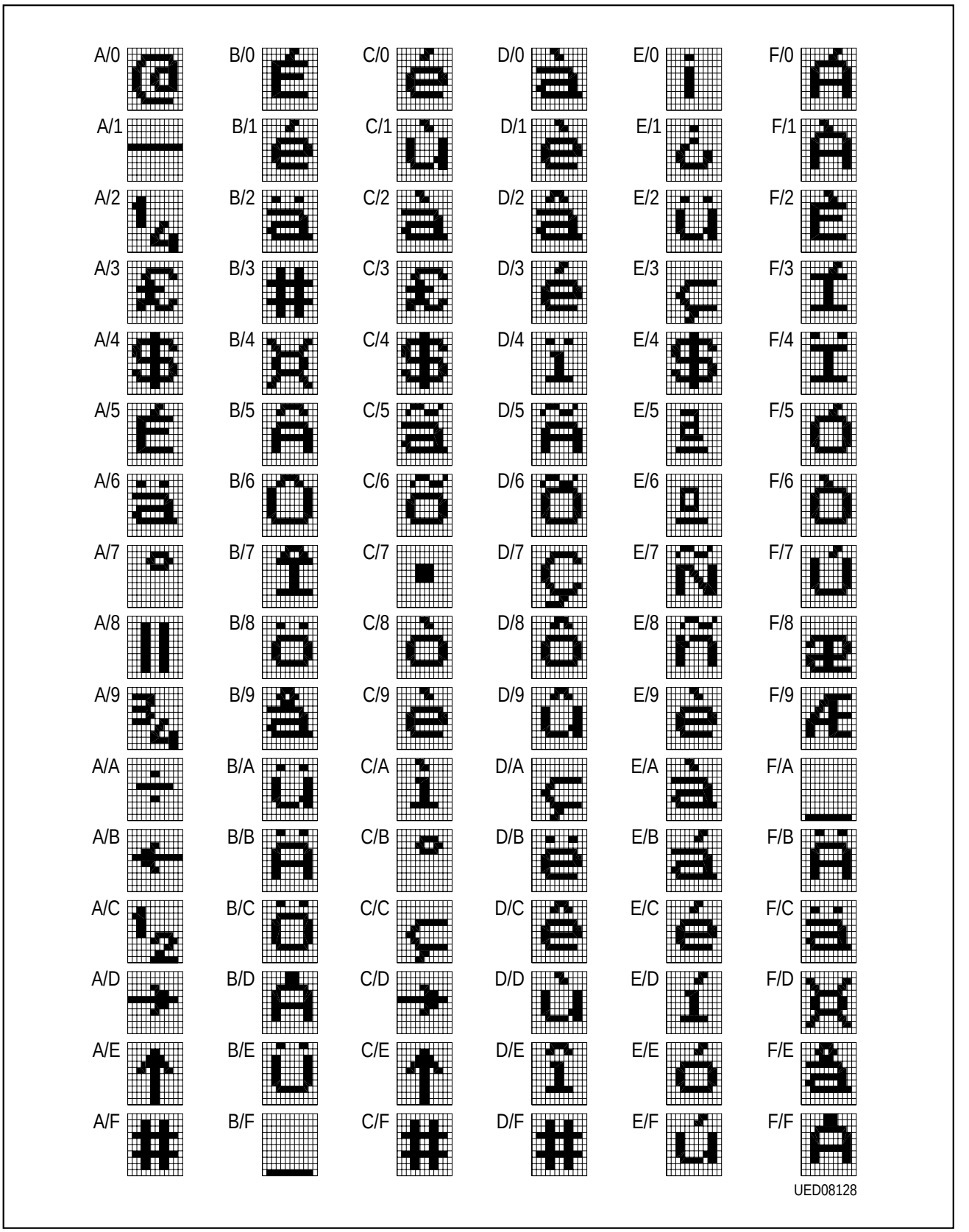


Figure 7
Character Set West Europe

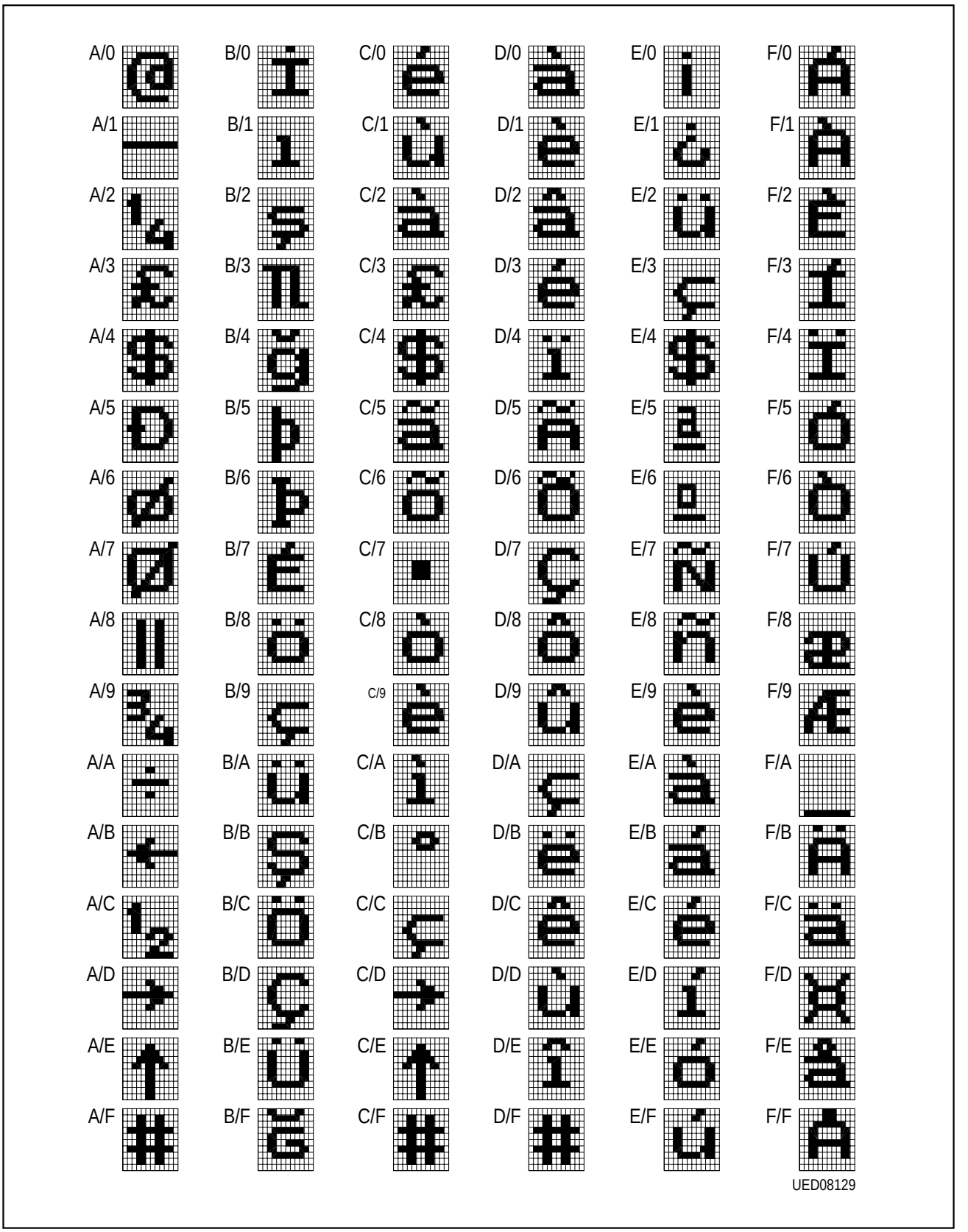


Figure 8
Character Set West Europe (Turkish)

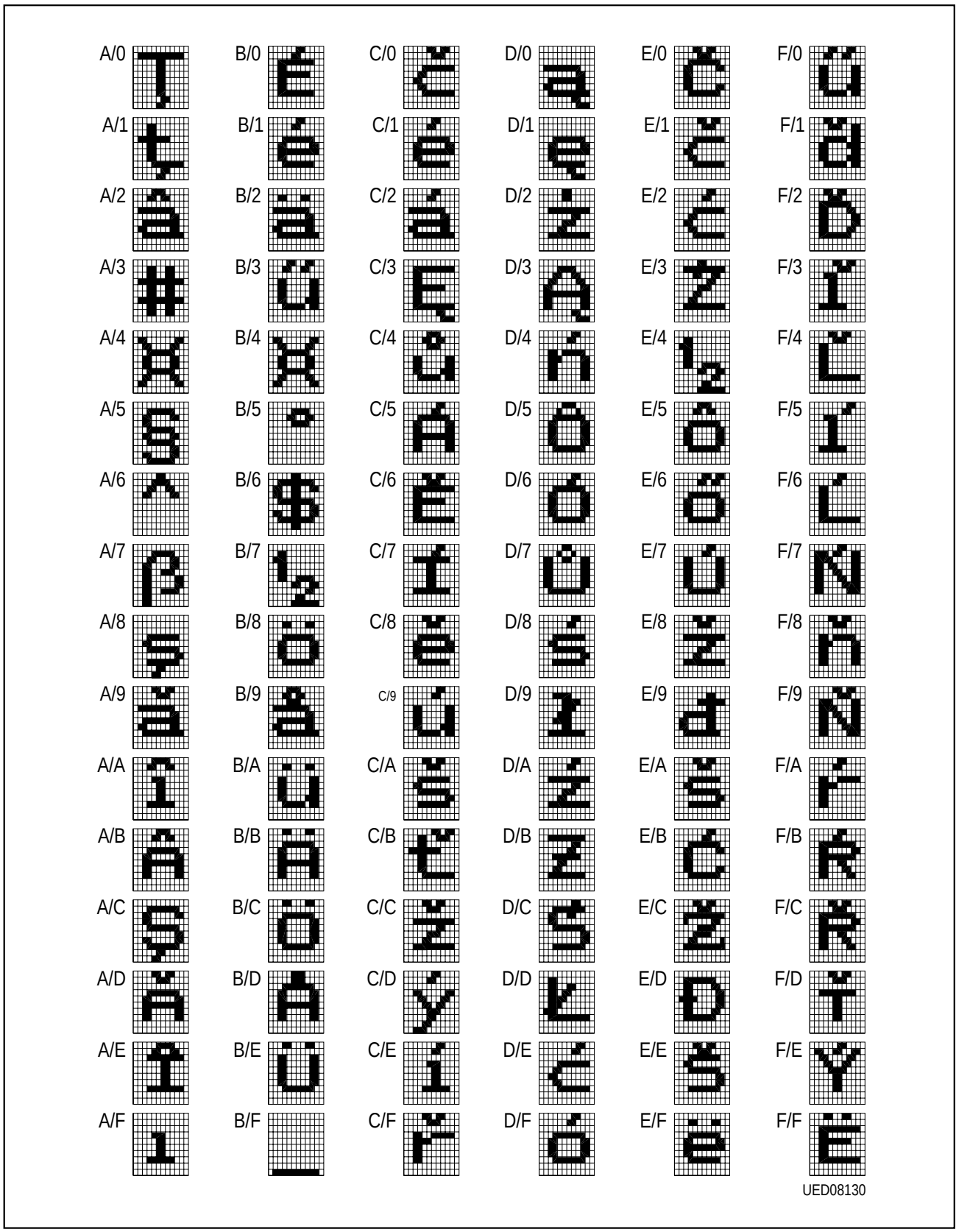


Figure 9
Character Set East Europe

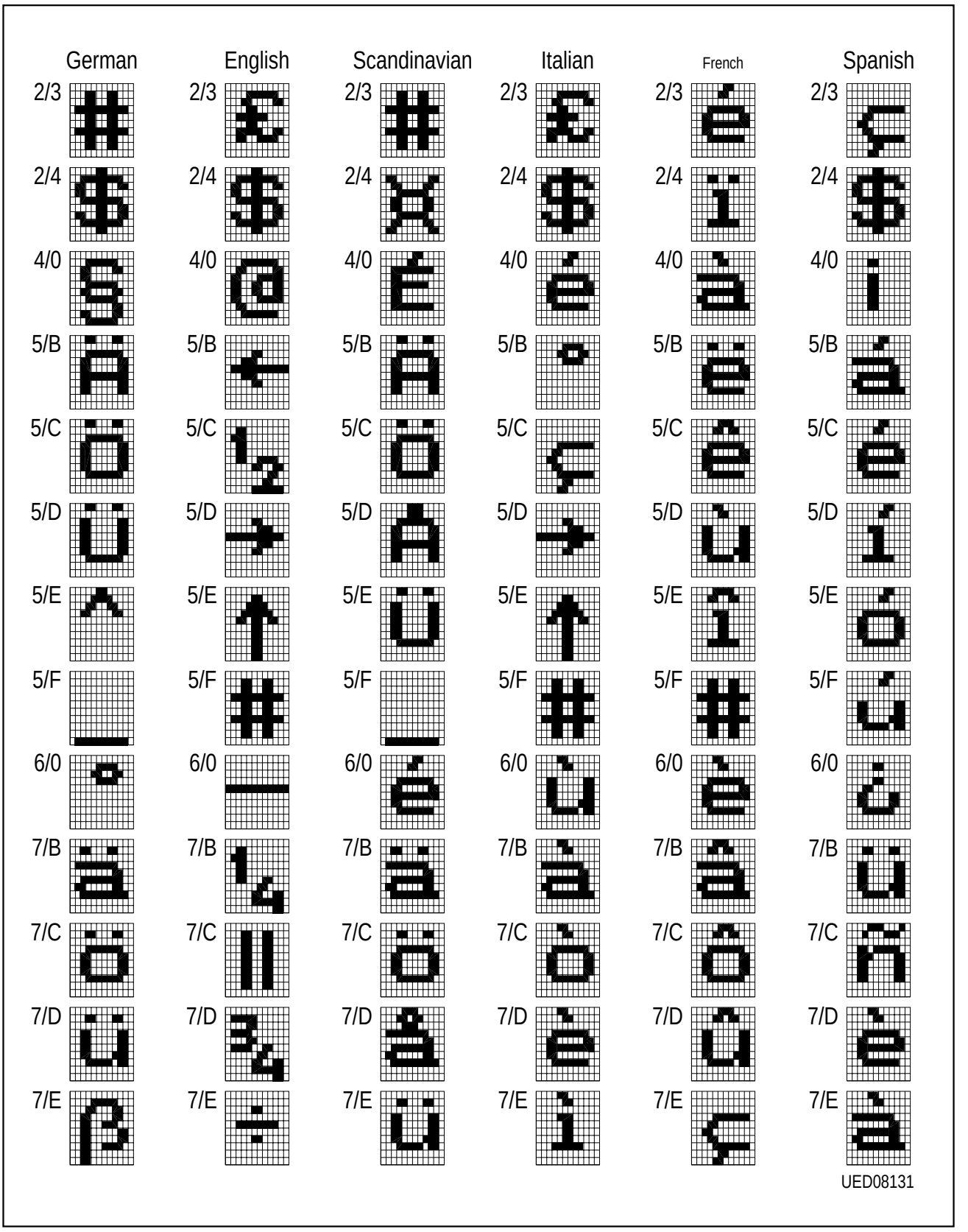


Figure 10
National Option Characters I

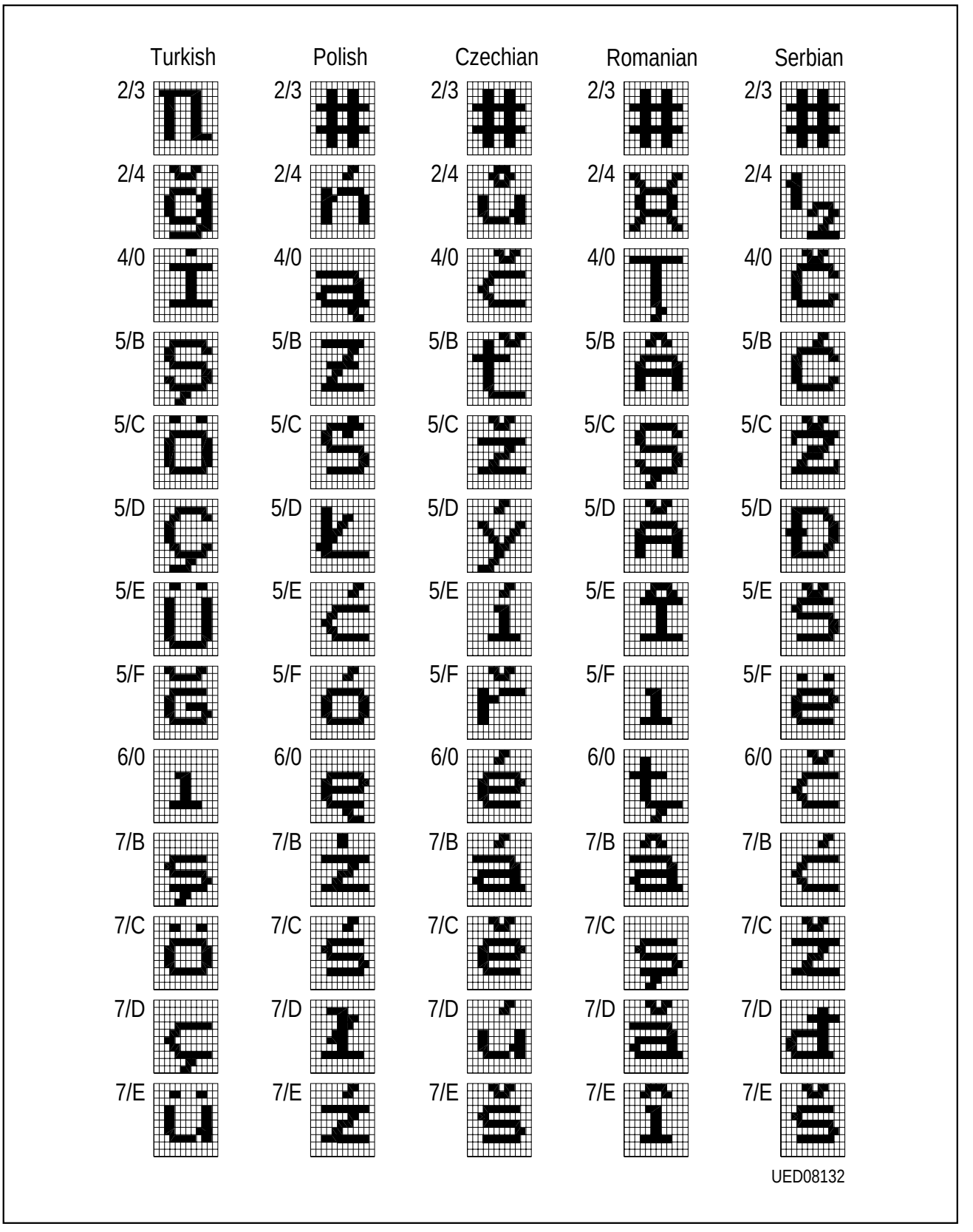


Figure 11
National Option Characters II

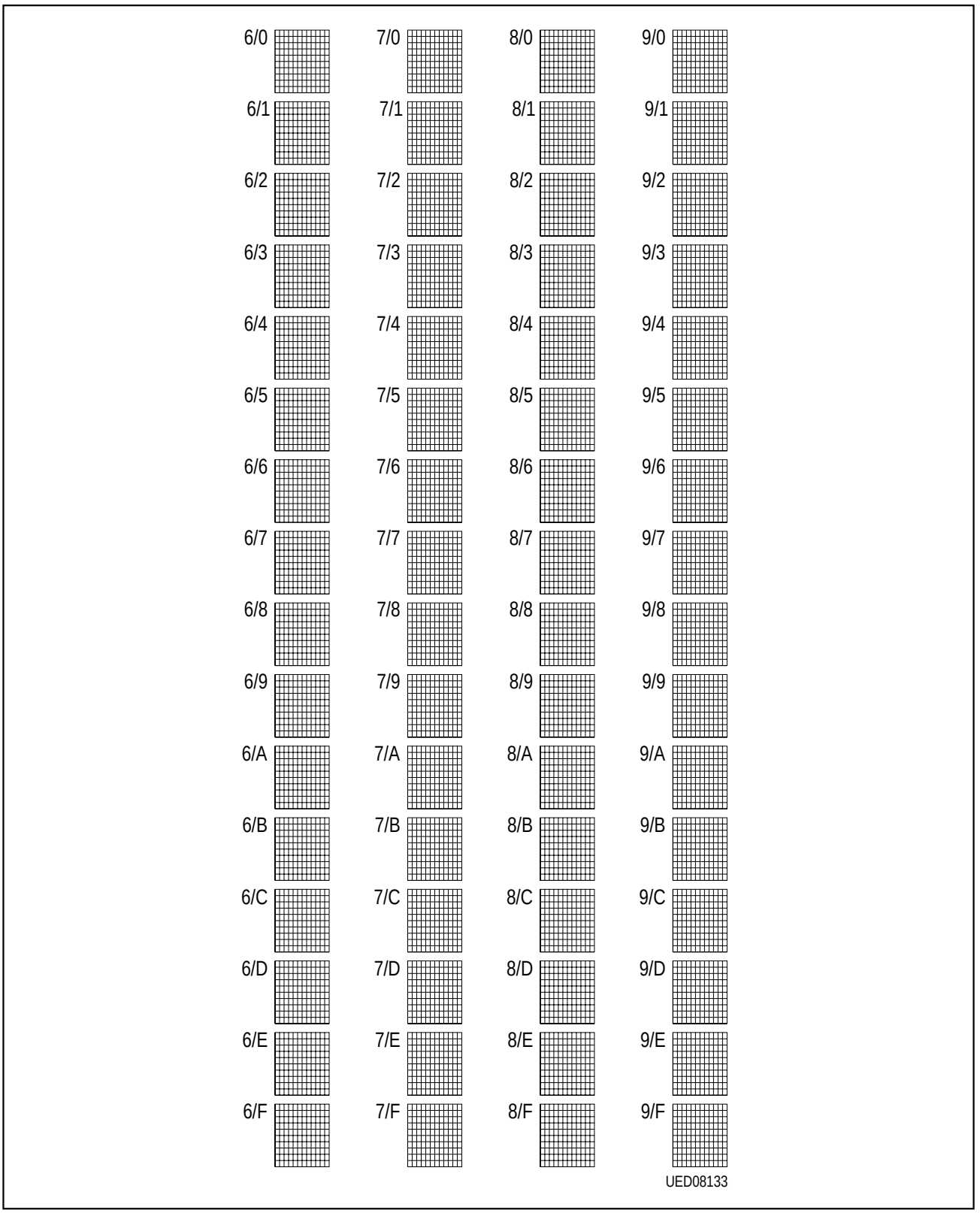


Figure 12
OSD Characters Set (these characters are customized and thus left blank on this page)

Note: Characters ... to ... can only be used inside an OSD box.

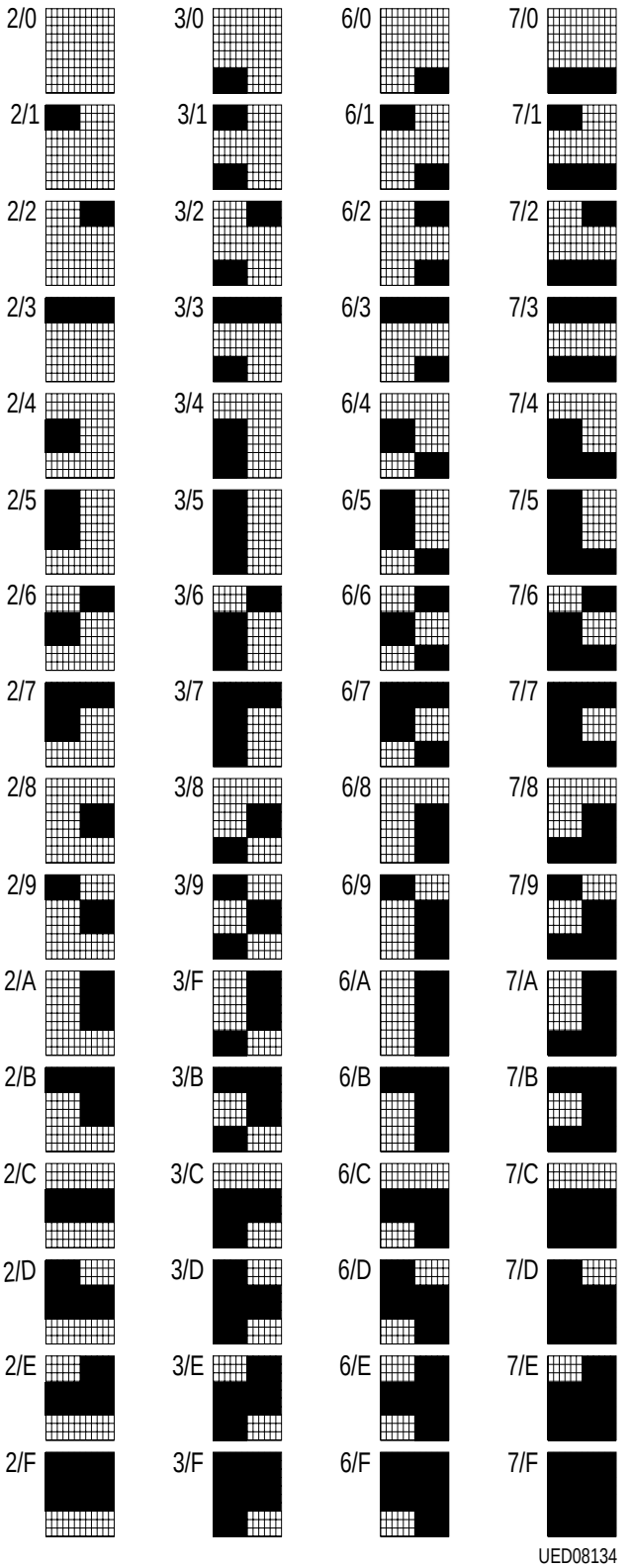


Figure 13
Graphics Character Set

6.3 Microcontroller

6.3.1 Architecture

The CPU manipulates operands in two memory spaces: the program memory space, and the data memory space. The program memory address space is provided to accommodate relocatable code.

The data memory address space is divided into the 256-byte internal data RAM, XRAM (extended data memory, accessible with MOVX-instructions) and the 128-byte Special Function Register (SFR) address spaces. Four register banks (each bank has eight registers), 128 addressable bits, and the stack reside in the internal data RAM. The stack depth is limited only by the available internal data RAM. It's location is determined by the 8-bit stack pointer. All registers except the program counter and the four 8-register banks reside in the special function register address space. These memory mapped registers include arithmetic registers, pointers, I/O-ports, registers for the interrupt system, timers, pulse width modulator and serial channel. Many locations in the SFR-address space are addressable as bits.

Note that reading from unused locations within data memory will yield undefined data.

Conditional branches are performed relative to the 16 bit program counter. The register-indirect jump permits branching relative to a 16-bit base register with an offset provided by an 8-bit index register. Sixteen-bit jumps and calls permit branching to any location in the memory address space.

The processor as five methods for addressing source operands: register, direct, register-indirect, immediate, and base-register plus index-register indirect addressing.

The first three methods can be used for addressing destination operands. Most instructions have a "destination, source" field that specifies the data type, addressing methods and operands involved. For operations other than moves, the destination operand is also a source operand.

Registers in the four 8-register banks can be accessed through register, direct, or register-indirect addressing; the lower 128 bytes of internal data RAM through direct or register-indirect addressing, the upper 128 bytes of internal data RAM through register-indirect addressing; and the special function registers through direct addressing. Look-up tables resident in program memory can be accessed through base-register plus index-register indirect addressing.

6.3.1.1 CPU-Hardware

Instruction Decoder

Each program instruction is decoded by the instruction decoder. This unit generates the internal signals that control the functions of each unit within the CPU-section. These signals control the sources and destination of data, as well as the function of the Arithmetic/Logic Unit (ALU).

Program Control Section

The program control section controls the sequence in which the instructions stored in program memory are executed. The conditional branch logic enables conditions internal and external to the processor to cause a change in the sequence of program execution. The 16-bit program counter holds the address of the instruction to be executed. It is manipulated with the control transfer instructions listed in **Chapter “Instruction Set” on page 116**.

Internal Data RAM

The internal data RAM provides a 256-byte scratch pad memory, which includes four register banks and 128 direct addressable software flags. Each register bank contains registers R0 – R7. The addressable flags are located in the 16-byte locations starting at byte address 32 and ending with byte location 47 of the RAM-address space.

In addition to this standard internal data RAM the processor contains an extended internal RAM. It can be considered as a part of an external data memory. It is referenced by MOVX-instructions (MOVX A, @DPTR), the memory map is shown in **Figure 21**.

Arithmetic/Logic Unit (ALU)

The arithmetic section of the processor performs many data manipulation functions and includes the Arithmetic/Logic Unit (ALU) and the A, B and PSW-registers. The ALU accepts 8-bit data words from one or two sources and generates an 8-bit result under the control of the instruction decoder. The ALU performs the arithmetic operations of add, subtract, multiply, divide, increment, decrement, BCD-decimal-add-adjust and compare, and the logic operations of and, or, exclusive-or, complement and rotate (right, left, or nibble swap).

The A-register is the accumulator, the B-register is dedicated during multiply and divide and serves as both a source and a destination. During all other operations the B-register is simply another location of the special function register space and may be used for any purpose.

Boolean Processor

The Boolean processor is an integral part of the processor architecture. It is an independent bit processor with its own instruction set, its own accumulator (the carry flag) and its own bit- addressable RAM and I/O. The bit manipulation instructions allow the direct addressing of 128 bits within the internal data RAM and several bits within the special function registers. The special function registers which have addresses exactly divisible by eight contain directly addressable bits.

The Boolean processor can perform, on any addressable bit, the bit operations of set, clear, complement, jump-if-set, jump-if-not-set, jump-if-set then-clear and move to/from carry. Between any addressable bit (or its complement) and the carry flag it can perform the bit operation of logical AND or logical OR with the result returned to the carry flag.

Program Status Word Register (PSW)

The PSW-flags record processor status information and control the operation of the processor. The carry (CY), auxiliary carry (AC), two user flags (F0 and F1), register bank select (RS0 and RS1), overflow (OV) and parity (P) flags reside in the program status word register. These flags are bit-memory-mapped within the byte-memory-mapped PSW. The CY, AC, and OV flags generally reflect the status of the latest arithmetic operations. The CY-flag is also the Boolean accumulator for bit operations. The P-flag always reflects the parity of the A-register. F0 and F1 are general purpose flags which are pushed onto the stack as part of a PSW-save. The two register bank select bits (RS1 and RS0) determine which one of the four register banks is selected as follows:

Table 6
Program Status Word Register

RS1	RS0	Register Bank	Register Location
0	0	0	00_H – 07_H
0	1	1	08_H – $0F_H$
1	0	2	10_H – 17_H
1	1	3	18_H – $1F_H$

Program Status Word PSW

Program Status Word **PSW** **SFR-Address $D0_H$**

(MSB) (LSB)

CY	AC	F0	RS1	RS0	OV	F1	P

Stack Pointer (SP)

The 8-bit stack pointer contains the address at which the last byte was pushed onto the stack. This is also the address of the next byte that will be popped. The SP is incremented during a push. SP can be read or written to under software control. The stack may be located anywhere within the internal data RAM address space and may be as large as 256 bytes.

Data Pointer Register (DPTR)

The 16-bit Data Pointer Register DPTR is the concatenation of registers DPH (high-order byte) and DPL (low-order byte). The DPTR is used in register-indirect addressing to move program memory constants and to access the extended data memory. DPTR may be manipulated as one 16-bit register or as two independent 8-bit registers DPL and DPH.

Eight data pointer registers are available, the active one is selected by a special function register (DPSEL).

Port 0, Port 1, Port 2, Port 3, Port 4

The five ports provide 26 I/O-lines and 5 input-lines to interface to the external world. All five ports are both byte and bit addressable. Port 0 is used for binary l/O and as clock and data line of a software driven I^2C bus. Port 1 provides eight PWM- output channels as alternate functions while port 2.0 - 2.3 are digital or analog inputs. Port 3 contains special control signals. Port 4 will usually be selected as memory extension interface (ROM-less version only).

Interrupt Logic

Controlled by three special function registers (IE, IP0 and IP1) the interrupt logic provides several interrupt vectors. Each of them may be assigned to high or low priority (see **Chapter “Interrupt System” on page 62**).

Timer/Counter 0/1

Two general purpose 16-bit timers/counters are controlled by the special function registers TMOD and TCON (see **Chapter “General Purpose Timers/Counters” on page 80**).

Serial Interface

A full duplex serial interface is provided where one of three operation modes may be selected. The serial interface is controlled by two special function registers (SCON, SBUF) as described in **Chapter “Serial Interface” on page 91**.

Watchdog Timer

For software- and hardware security, a watchdog timer is supplied, which resets the processor, if not cleared by software within a maximum time period.

Pulse Width Modulation Unit

Up to six lines of port 1 may be used as 8-bit PWM-outputs and two lines of port 1 may be used as 14-bit PWM-output. The PWM-logic is controlled by registers PWCOMP0 … 7, PWCL, PWCH, PWME, PWEXT6, PWEXT7 (see **Chapter “Pulse Width Modulation Unit (PWM)” on page 106**).

Capture Compare Timer

For easy decoding of infrared remote control signals, a dedicated timer is available (see **Chapter “Capture Compare Timer” on page 90**).

6.3.1.2 CPU-Timing

Timing generation is completely self-contained, except for the frequency reference which can be a crystal or external clock source. The on-board oscillator is a parallel anti-resonant circuit. There is a divide-by-6 internal timing which leads to a minimum instruction cycle of 0.33 µs with an 18-MHz crystal. The XTAL2-pin is the output of a high-gain amplifier, while XTAL1 is its input. A crystal connected between XTAL1 and XTAL2 provides the feedback and phase shift required for oscillation.

A machine cycle consists of 6 oscillator periods (software selectable). Most instructions execute in one cycle. MUL (multiply) and DIV (divide) are the only instructions that take more than two cycles to complete. They take four cycles.

To reduce the power consumption, the internal clock frequency can be divided by two, which slows down the processor operations.

This slow down mode is entered by setting SFR-Bit CDC in register AFR.

Note: All timing values and diagrams in this specification refer to an inactivated clock divider (CDC = 0).

Note: Slow down mode should only be used if teletext reception and the display are disabled. Otherwise processing of the incoming text data might be incomplete and the display structure will be corrupted.

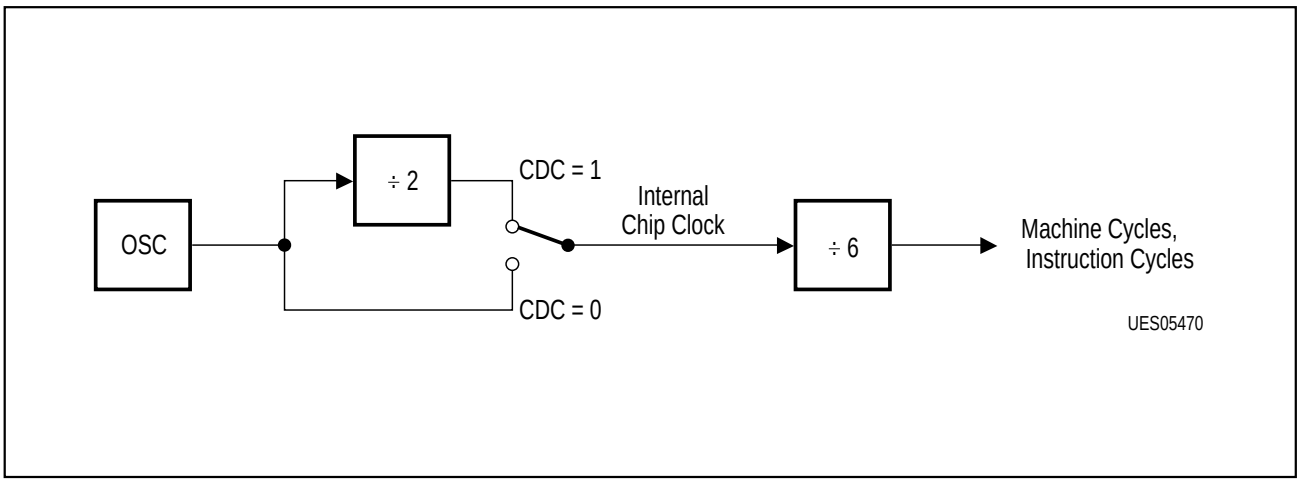


Figure 14
CPU-Timing

Note: For CDC see ***Chapter "Advanced Function Register" on page 115****.*

6.3.1.3 Addressing Modes

There are five general addressing modes operating on bytes. One of these five addressing modes, however, operates on both bytes and bits:

– Register
– Direct (both bytes and bits)
– Register indirect
– Immediate
– Base-register plus index-register indirect

The following section summarizes, which memory spaces may be accessed by each of the addressing modes:

Register Addressing
R0 – R7
ACC, B, CY (bit), DPTR

Direct Addressing
RAM (low part)
Special Function Registers

Register-Indirect Addressing
RAM (@R1, @R0, SP)

Immediate Addressing
Program Memory

Base-Register plus Index-Register Indirect Addressing
Program Memory (@DPTR + A, @PC + A)

Register Addressing

Register addressing accesses the eight working registers (R0 – R7) of the selected register bank. The PSW-register flags RS1 and RS0 determine which register bank is enabled. The least significant three bits of the instruction opcode indicate which register is to be used. ACC, B, DPTR and CY, the Boolean processor accumulator, can also be addressed as registers.

Direct Addressing

Direct byte addressing specifies an on-chip RAM-location (only low part) or a special function register. Direct addressing is the only method of accessing the special function registers. An additional byte is appended to the instruction opcode to provide the memory location address. The highest-order bit of this byte selects one of two groups of addresses: values between 0 and 127 (00_H – $7F_H$) access internal RAM-locations, while values between 128 and 255 (80_H – $0FF_H$) access one of the special function registers.

Register-Indirect Addressing

Register-indirect addressing uses the contents of either R0 or R1 (in the selected register bank) as a pointer to locations in the 256 bytes of internal RAM. Note that the special function registers are not accessable by this method.

Execution of PUSH- and POP-instructions also use register-indirect addressing. The stack pointer may reside anywhere in internal RAM.

Immediate Addressing

Immediate addressing allows constants to be part of the opcode instruction in program memory.

An additional byte is appended to the instruction to hold the source variable. In the assembly language and instruction set, a number sign (#) precedes the value to be used, which may refer to a constant, an expression, or a symbolic name.

Base-Register plus Index Register-Indirect Addressing

Base-register plus index register-indirect addressing allows a byte to be accessed from program memory via an indirect move from the location whose address is the sum of a base register (DPTR or PC) and index register, ACC. This mode facilitates accessing to look-up-table resident in program memory.

6.3.2 Memory Organization

The processor memory is organized into two address spaces. The memory spaces are:

– Program memory address space
– 256 byte plus 128-byte internal data memory address space
– Extended internal data memory (XRAM) for storing teletext and display data.

A 16-bit program counter and a dedicated banking logic provide the processor with its 512-Kbyte addressing capabilities (for ROM-less versions, up to 19 address lines are available). The program counter allows the user to execute calls and branches to any location within the program memory space. There are no instructions that permit program execution to move from the program memory space to any of the data memory space.

6.3.2.1 Program Memory

Certain locations in program memory are reserved for specific programs. Locations 0000 through 0002 are reserved for the initialization program. Following reset, the CPU always begins execution at location 0000. Locations 0003 through 0051 are reserved for the seven interrupt-request service programs as indicated in **Table 7**.

Table 7

Source	Address
External Interrupt 0	03 (03_H)
Timer 0 Overflow	11 ($0B_H$)
External Interrupt 1	19 (13_H)
Timer 1 Overflow	27 ($1B_H$)
Serial Interface	35 (23_H)
Teletext Sync Signals	43 ($2B_H$)
Analog Digital Converter	51 (33_H)

Depending on the selected type, the user can access a part of the internal/external ROM for the application software. Please note that another part of the Program Memory is reserved for the TTX firmware.

Table 8

Type	Available User ROM Space		
SDA 5250	480	Kbyte	externally
SDA 5251	8	Kbyte	internally
SDA 5252	16	Kbyte	internally
SDA 5254	16	Kbyte	internally
SDA 5255	24	Kbyte	internally

Memory Extension (ROMless version only)

The processor is prepared to extend its external program memory space up to 512 Kbytes (**Figure 15** and **16**). For easy handling of existing software and assemblers this space is split into 8 banks of 64 Kbytes each. The extension concept, based on the standard 64 K addressing ability, is provided for high effective and easy memory access with minimum software overhead. There is also no need caring about bank organization during subroutine processing or interrupts. This is done through address bits A16 – 18, which are controlled by a special internal circuitry, performing a “delayed banking”. The operations to the extended memory spaces are controlled by two additional special function registers called MEX1 and MEX2 (**Figure 17**). The address bits A17 and A18 are implemented at port 4. Programs, using only 128-Kbytes program memory space, may switch the address function off by setting bits NB, IB and bits MB to ‘1’ followed by a LJMP. Then port 4 will work properly in port mode. Whenever full address mode is desired, port 4 bits have to be kept on ‘1’ (**Table 9**). After reset all CB are ‘0’ and P4 latches are set to ‘1’, resulting a ‘0’ at the port 4 pins.

Banking of Program Memory

After reset the bits for current bank (CB) and next bank (NB) are set to zero. This way the processor starts the same as any 8051 controller at address 00000_H. Whenever a jump to another bank is required, the software has to change the bits NB16 – 18 for initializing the bank exchange (bits CB16 – 18 are read only). After operating the next LJMP instruction the NB16 – 18 bits (next bank) are copied to CB16 – 18 (current bank) and will appear at A16 – 18. Only LJMP will do this.

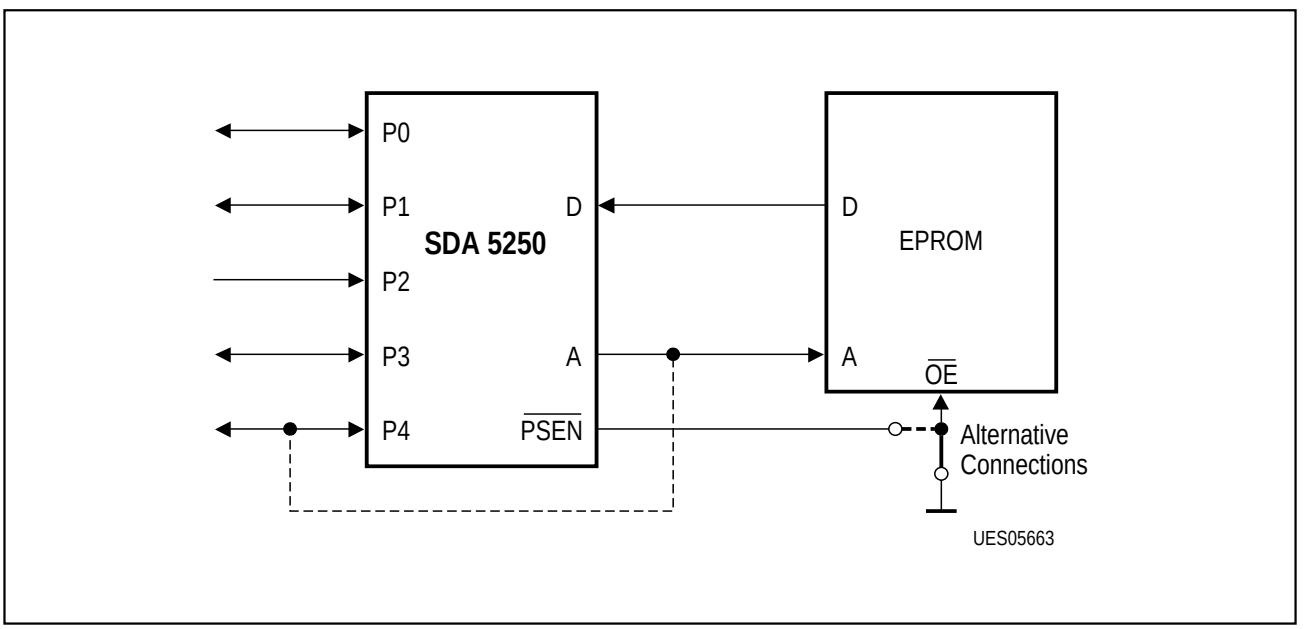


Figure 15
Connecting External Program Memory

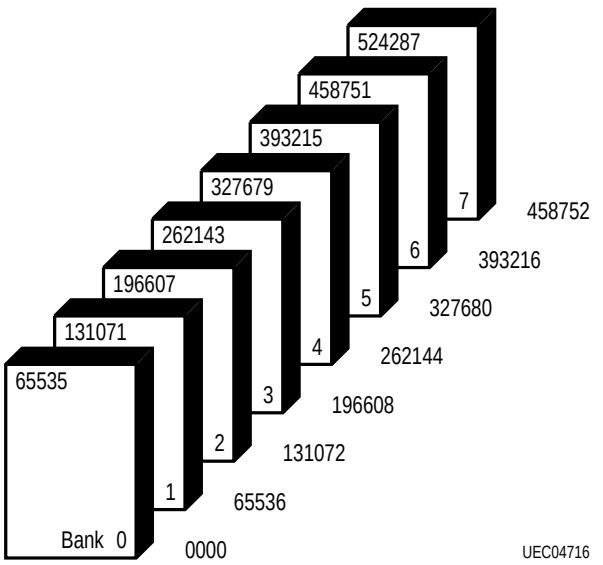


Figure 16
Bank Organization

MEX1 (94_H): Bank Control

7	6	5	4	3	2	1	0
–	CB18	CB17	CB16	–	NB18	NB17	NB16

MEX2 (95_H): Mode Control

7	6	5	4	3	2	1	0
MM	MB18	MB17	MB16	SF	IB18	IB17	IB16

CB = Current Bank	Read only; CBx = Ax
NB = Next Bank	R/W
MM = Memory Mode	R/W; 1 = use MB
MB = Memory Bank	R/W
SF = Stack Full	Read only; 1 = full
IB = Interrupt Bank	R/W

Figure 17
Register Bits MEX1 / MEX2

Table 9
Port 4 Configuration

CB	P4 Latch	P4 Out	Comment
0	0	0	x
0	1	0	Address
1	0	0	P4
1	1	1	Addr / P4

MOVC-Handling

MOVC-instructions may operate in two different modes, that are selected by bit MM in MEX2. On MM = 0 MOVC will access the current bank. On MM = 1 the bits MB16 – 18 will appear at A16 – A18 during MOVC.

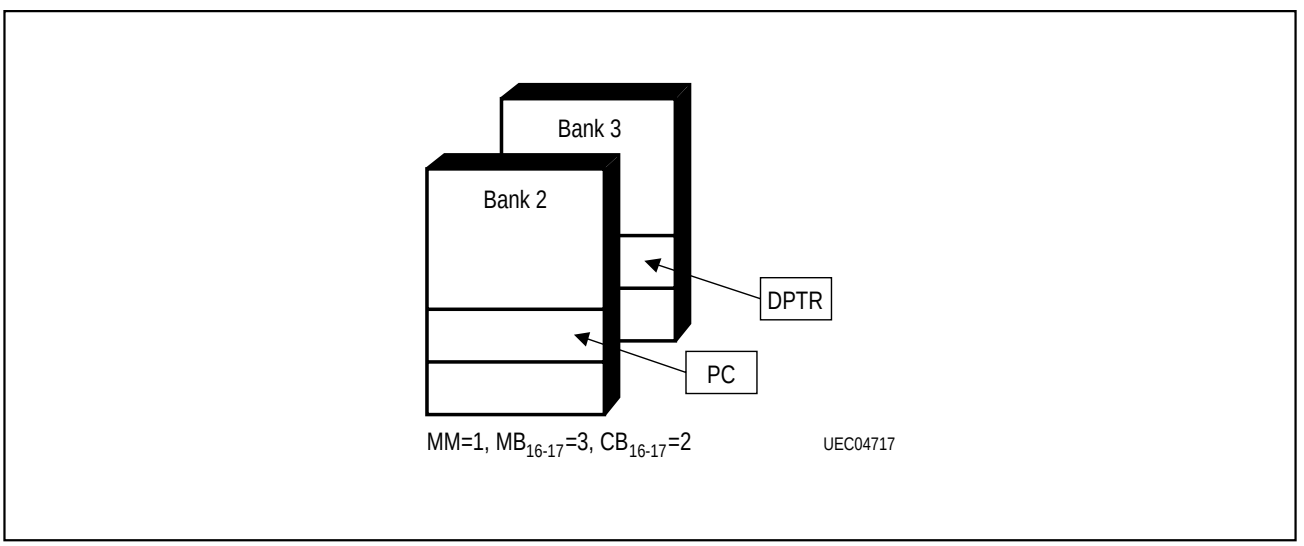


Figure 18
PC and DPTR on Different Banks

CALLs and Interrupts

For flexible use of CALL and interrupts the control logic holds an own 32 levels-six-bit-stack. Whenever a LCALL or ACALL occurs, CB16 – 18 and NB16 – 18 (MEX1) is copied to this stack and the memory extension stackpointer is incremented. Then NB16 – 18 is copied to CB16 – 18. Leaving subroutines through RET or RETI decrements the stack pointer and reads the old NB and CB contents from the stack. All six bits are required for saving to prevent conflicts on interrupt events. One additional feature simplifies the handling of interrupts: on occurrence the bits IB16 – 18 within MEX2 are copied to CB16 – 18 and NB16 – 18 after pushing their old contents on the stack. This way programmers can place their ISR (Interrupt Service Routine) on specific banks. After reset MM, MB16 – 18 and IB16 – 18 are set to zero.

In order to prevent loss of program control during deep subroutine nesting a warning bit “SF” (Stack Full) is set in MEX2 whenever a memory extension stack depth overflow is imminent. For example **Figure 19** shows the data flows at the memory extension stack during a LCALL. All three bits of NB are copied to the position CB and NB of the next higher stack level (now the current MEX1) while the last CB and NB are held on the stack. Returning from subroutine through RET the memory extension stack pointer decrements and CB and NB of MEX1 has the same contents as before LCALL.

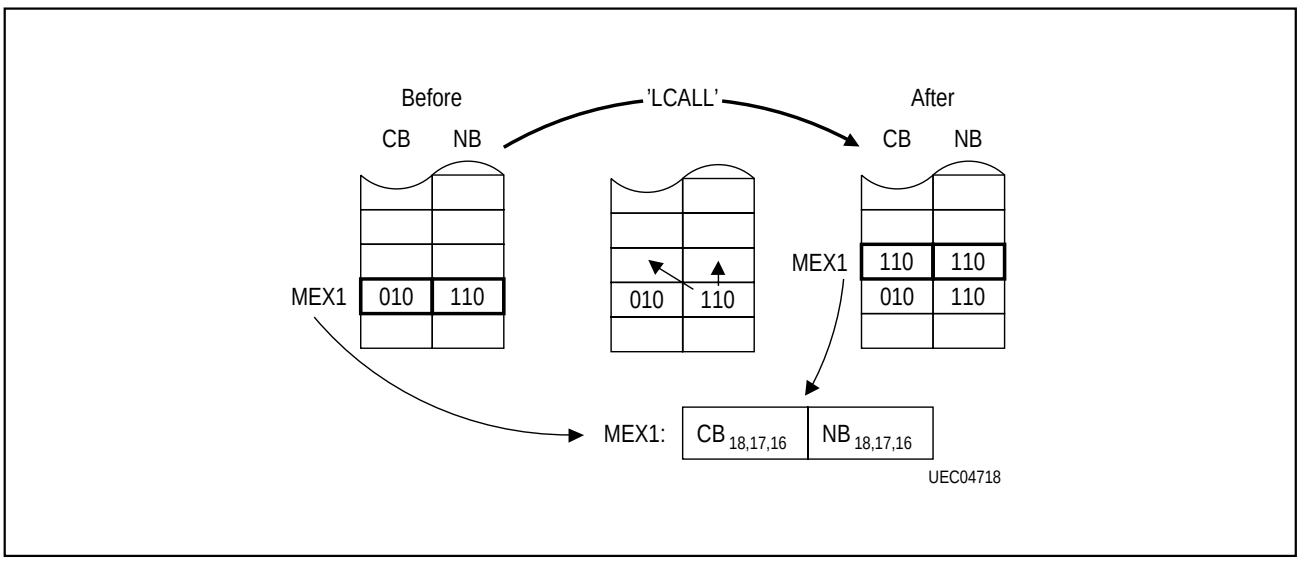


Figure 19
Processing LCALL (same as ACALL)

Examples

The standard sequence jumping from one bank to another is simply preceding a “MOV MEX1,#”- instruction to an “LJMP / LCALL” as shown in **Figure 19**. To operate programs up to 512 Kbytes with standard assemblers or from C the program can be split into sections, modules or files, that will each run in their own bank. Referencing banks to each other (jumps, calls, data moves) may be done by a simple preprocessing of the source programs or object files. Users, going to program a 512-Kbyte EPROM in assembler, may proceed like this:

1. Build up to eight assembler source files (max. 64 K), inter bank operations will refer to dummy labels.
2. Do assembler runs on each block and generate label lists.
3. Preprocessing: substitute the inter bank labels in the source files with absolute 64 K addresses.
4. Second and final assembler runs on each block, generate Hex files.
5. Append the Hex files in right order.
6. Program an EPROM.

More comfortable programming, e. g. based on C-programs, require similar processing of the source programs or object files with respect to special considerations of the compiler.

Figure 20 shows an assembler program run, performing the following actions:

1. Start at bank 0 at 00000.
2. Set ISR-page to bank 2.
3. Jump to bank 1 at address 25.
4. Being interrupted to bank 2 ISR.
5. Call a subprogram at bank 2 address 43.
6. After return read data from bank 2.

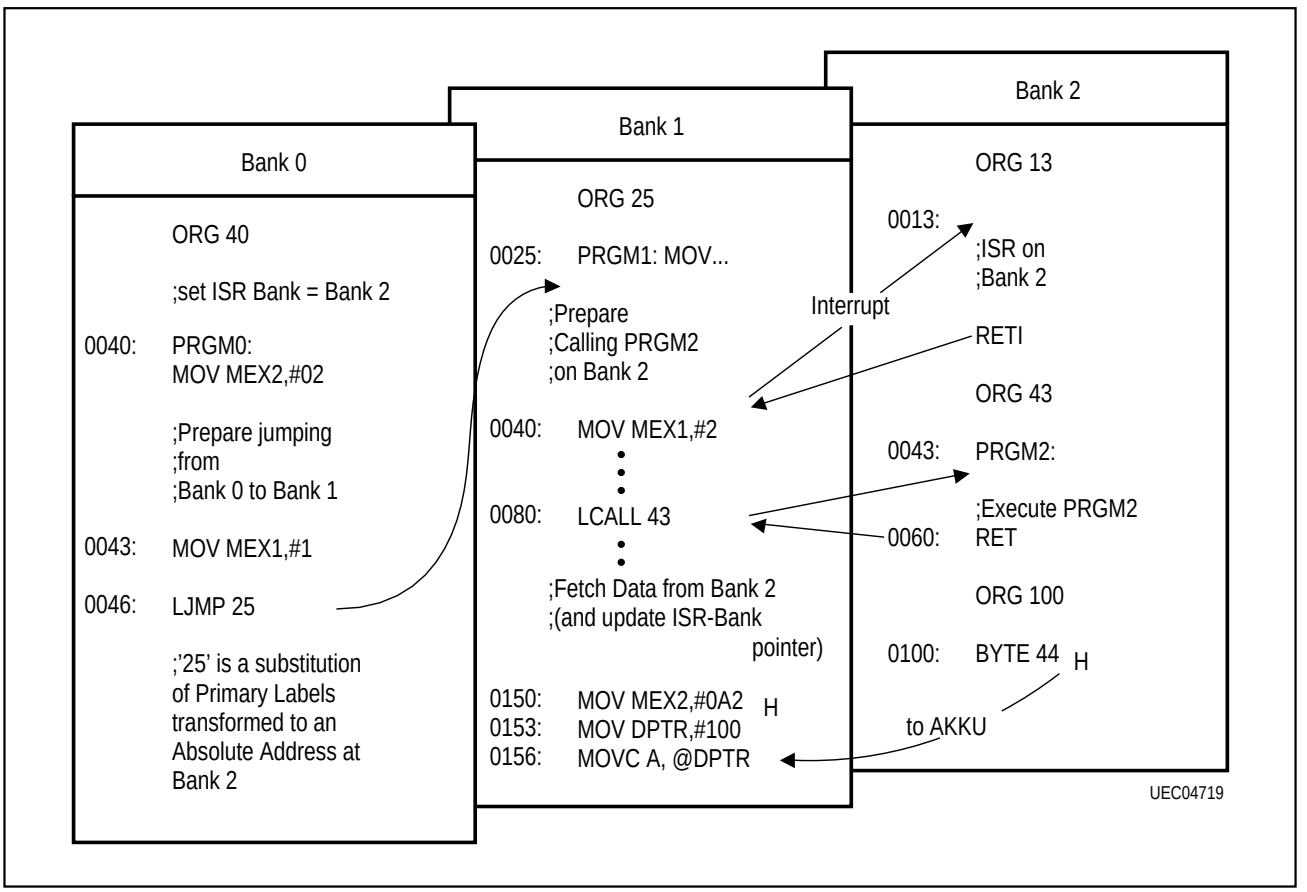


Figure 20
Program Example

6.3.2.2 Internal Data RAM

The internal data memory is divided into four blocks: the lower 128 byte of RAM, the upper 128 byte of RAM, the 128-byte Special Function Register (SFR) area and the up to 10 Kbyte additional RAM (**Figure 21**). Because the upper RAM-area and the SFR-area share the same address locations, they are accessed through different addressing modes.

The internal data RAM-address space is 0 to 255. Four banks of eight registers each occupy locations 0 through 31. Only one of these banks may be enabled at a time through a two-bit field in the PSW. In addition, 128-bit locations of the on-chip RAM are accessible through direct addressing.

These bits reside in internal data RAM at byte locations 32 through 47, as shown in **Table 11**. The lower 128 bytes of internal data RAM can be accessed through direct or register-indirect addressing, the upper 128 bytes of internal data RAM through register-indirect addressing and the special function registers through direct addressing.The stack can be located anywhere in the internal data RAM-address space. The stack depth is limited only by the available internal data RAM, thanks to an 8-bit relocatable stack pointer. The stack is used for storing the program counter during subroutine calls and may also be used for passing parameters. Any byte of internal data RAM or special function registers accessible through direct addressing can be pushed/popped.

An additional on-chip RAM-space called “XRAM” extends the internal RAM-capacity. The up to 10 Kbytes of XRAM are accessed by MOVX @DPTR. XRAM is located in the upper area of the address space. 1 Kbyte of the XRAM, called VBI Buffer, is reserved for storing teletext data and another up to 8 Kbyte of the XRAM, called Display Chapters, are reserved for storing up to 8 display chapters (see **Figure 21**). Unused memory area of the VBI Buffer and the Display Chapters can be used by the controller as general RAM space.

Table 10
XRAM Address Space

Function	Byte Address (hex.)
VBI Buffer	F400 - F7FF
Display Chapter 0 - 7	C000 - DFFF(1)
CPU XRAM	F800 - FBFF(2)

(1) SDA 5251, SDA 5252 C000 - C3FF only
(2) SDA 5250, SDA 5254 and SDA 5255 only

6.3.2.3 Special Function Registers

The special function register address space resides between addresses 128 and 255. All registers except the program counter and the four banks of eight working registers reside here. Memory mapping the special function registers allows them to be accessed as easily as the internal RAM. As such, they can be operated on by most instructions. A complete list of the special function registers is given in **Table 13**.

In addition, many bit locations within the special function register address space can be accessed using direct addressing. These direct addressable bits are located at byte addresses divisible by eight as shown in **Table 12**.

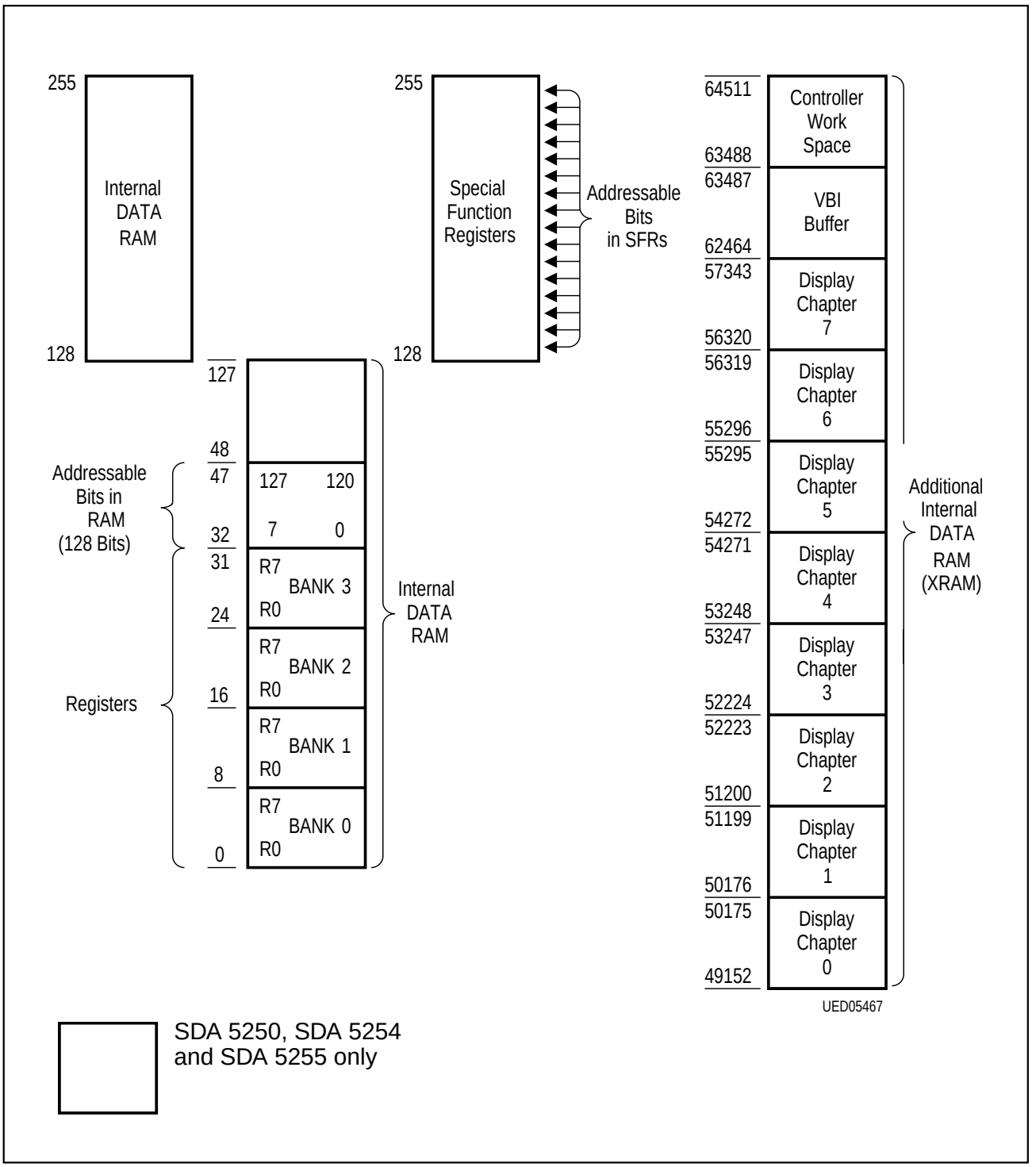


Figure 21
Internal Data Memory Address Space

Table 11
Internal RAM-Bit Addresses

RAM Byte	(MSB)							(LSB)	
256									FF_H
≈									≈
47	7F	7E	7D	7C	7B	7A	79	78	$2F_H$
46	77	76	75	74	73	72	71	70	$2E_H$
45	6F	6E	6D	6C	6B	6A	69	68	$2D_H$
44	67	66	65	64	63	62	61	60	$2C_H$
43	5F	5E	5D	5C	5B	5A	59	58	$2B_H$
42	57	56	55	54	53	52	51	50	$2A_H$
41	4F	4E	4D	4C	4B	4A	49	48	29_H
40	47	46	45	44	43	42	41	40	28_H
39	3F	3E	3D	3C	3B	3A	39	38	27_H
38	37	36	35	34	33	32	31	30	26_H
37	2F	2E	2D	2C	2B	2A	29	28	25_H
36	27	26	25	24	23	22	21	20	24_H
35	1F	1E	1D	1C	1B	1A	19	18	23_H
34	17	16	15	14	13	12	11	10	22_H
33	0F	0E	0D	0C	0B	0A	09	08	21_H
32	07	06	05	04	03	02	01	00	20_H
31	Bank 3								$1F_H$
24									18_H
23	Bank 2								17_H
16									10_H
15	Bank 1								$0F_H$
8									08_H
7	Bank 0								07_H
0									00_H

Table 12
Special Function Register Bit Address Space

Direct Byte Address	Bit Address								Hardware Register Symbol
$F8_H$	FF	FE	FD	FC	FB	FA	F9	F8	PWME
$F0_H$	F7	F6	F5	F4	F3	F2	F1	F0	B
$E8_H$	–	–	–	–	–	–	E9	E8	P4
$E0_H$	E7	E6	E5	E4	E3	E2	E1	E0	ACC
$D8_H$	DF	DE	DD	DC	DB	–	D9	D8	ADCON
$D0_H$	D7	D6	D5	D4	D3	D2	D1	D0	PSW
$C8_H$	–	CE	CD	CC	CB	CA	C9	C8	TTXSIR
$C0_H$	C7	C6	C5	C4	C3	C2	C1	C0	ACQSIR
$B8_H$	–	–	–	–	–	–	–	–	
$B0_H$	B7	B6	B5	B4	B3	B2	B1	B0	P3
$A8_H$	AF	AE	AD	AC	AB	AA	A9	A8	IE
$A0_H$	–	–	–	–	A3	A2	A1	A0	P2
98_H	9F	9E	9D	9C	9B	9A	99	98	SCON
90_H	97	96	95	94	93	92	91	90	P1
88_H	8F	8E	8D	8C	8B	8A	89	88	TCON
80_H	87	86	85	84	83	82	81	80	P0

Table 13
Special Function Register Overview

Special Function Register Description	Symbolic Name	Address Location (hex.)	Address Location (dec.)	Bit Address MSB … LSB (hex.)	Initial Value after Reset (hex./bin.)
Arithmetic Registers					
Accumulator	ACC, A	E0	224	E7 - E0	00
B-Register	B	F0	240	F7 - F0	00
Program Status Word	PSW	D0	208	D7 - D0	00
System Control Registers					
Stack Pointer	SP	81	129	–	07
Data Pointer (high byte)	DPH	83	131	–	00
Data Pointer (low byte)	DPL	82	130	–	00
Data Pointer Select	DPSEL	A2	162	–	xxxxx000
Power Control	PCON	87	135		000xxx00
I/O-Port Registers					
Port 0	P0	80	128	87 - 80	FF
Port 1	P1	90	144	97 - 90	FF
Port 2	P2	A0	160	A3 - A0	FF
Port 3	P3	B0	176	B7 - B0	FF
Port 4	P4	E8	232	E9 - E8	xxxxxx00
Interrupt Control Registers					
Interrupt Enable Flags	IE	A8	168	AF - A8	00
Interrupt Priority Flags	IP0	A9	169	–	00
Interrupt Priority Flags	IP1	AA	170	–	00
Interrupt Control	IRCON	A8	171	–	xxxx0101
Timer 0/1 Registers					
Timer 0/1 Mode Register	TMOD	89	137	–	00
Timer 0/1 Control Register	TCON	88	136	8F - 88	00
Timer 1 (high byte)	TH1	8D	141	–	00
Timer 0 (high byte)	TH0	8C	140	–	00
Timer 1 (low byte)	TL1	8B	139	–	00
Timer 0 (low byte)	TL0	8A	138	–	00
Watchdog Timer Registers					
Watchdog Control Register	WDCON	A7	167	–	00
Watchdog Reload Register	WDTREL	86	134	–	00
Watchdog Low Byte	WDTL	84	132	–	00
Watchdog High Byte	WDTH	85	133	–	00
Capture Compare Timer Registers	RELL	E1	225	–	xx
	RELH	E2	226	–	xx
	CAPL	E3	227	–	xx
	CAPH	E4	228	–	xx
	IRTCON	E5	229	–	00

Table 13
Special Function Register Overview (cont'd)

Special Function Register Description	Symbolic Name	Address Location (hex.)	Address Location (dec.)	Bit Address MSB … LSB (hex.)	Initial Value after Reset (hex./bin.)
Analog Digital Converter					
ADC-Control Register	ADCON	D8	216	DF - D8	00
ADC-Data Register	ADDAT	D9	217	–	00
ADC-Start Register	DAPR	DA	218	–	xx
Pulse Width Modulator Registers					
Enable Register	PWME	F8	248	FF - F8	00
Counter Register (low byte)	PWCL	F7	247	–	00
Counter Register (high byte)	PWCH	F9	249	–	00
Compare Register 0	PWCOMP0	F1	241	–	FF
Compare Register 1	PWCOMP1	F2	242	–	FF
Compare Register 2	PWCOMP2	F3	243	–	FF
Compare Register 3	PWCOMP3	F4	244	–	FF
Compare Register 4	PWCOMP4	F5	245	–	FF
Compare Register 5	PWCOMP5	F6	246	–	FF
PWM 14 Compare Reg. 0	PWCOMP6	FB	251	–	FF
PWM 14 Extension Reg. 0	PWEXT6	FA	250	–	FF
PWM 14 Compare Reg. 1	PWCOMP7	FD	253	–	FF
PWM 14 Extension Reg. 1	PWEXT7	FC	252	–	FF
Serial Interface Registers					
Serial Control Register	SCON	98	144	9F - 98	00
Serial Data Register	SBUF	99	145	–	xx
Advanced Function Register	AFR	A6	166	–	00xxxxxx
Slicer Control Registers					
Acq. Sync Interrupt Register	ACQSIR	C0	192	C7 - C0	00
Acquisition Mode Register 1	ACQMS_1	C1	193	–	00
Acquisition Mode Register 2	ACQMS_2	C2	194	–	00

Table 13
Special Function Register Overview (cont'd)

Special Function Register Description	Symbolic Name	Address Location (hex.)	Address Location (dec.)	Bit Address MSB … LSB (hex.)	Initial Value after Reset (hex./bin.)
Display Control Registers					
Horizontal Delay	DHD	C3	195	–	00
Vertical Delay	DVD	C4	196	–	00
Transparent Control	DTCR	C5	197	–	00
Mode 1 Register	DMODE1	C6	198	–	00
Mode 2 Register	DMODE2	C7	199	–	00
Sync Interrupt Request Reg.	TTXSIR	C8	200	CF - C8	00
Language Control	LANGC	C9	201	–	00
Cursor Column Position	DCCP	CA	202	–	00
Cursor Row Position	DCRP	CB	203	–	00
Display Timing Control	DTIM	CC	204	–	00
Sandcastle Control	SCCON	CE	206	–	00
Display Mode	DMOD	D6	214	–	x0

6.3.3 Interrupt System

External events and the real-time on-chip peripherals require CPU-service asynchronous to the execution of any particular section of code. To couple the asynchronous activities of these functions to normal program execution, a sophisticated multiple-source, four-priority-level, nested interrupt system is provided. Interrupt response delay ranges from 0,89 µs to 2.33 µs when using an 18-MHz clock (see **Chapter "Advanced Function Register" on page 115**).

6.3.3.1 Interrupt Sources

The processor acknowledges interrupt requests from seven sources: two from external sources via the $\overline{INT0}$ and $\overline{INT1}$ pins, one from each of the two internal counters, one from the serial I/O-port, one from teletext sync signals and one from the analog digital converter. Each of the seven sources can be assigned to either of four priority levels and can be independently enabled and disabled. Additionally, all enabled sources can be globally disabled or enabled.

Interrupts result in a transfer of control to a new program location. Each interrupt vectors to a separate location in program memory for its service program. The program servicing the request begins at this address. The starting address (interrupt vector) of the interrupt service program for each interrupt source is shown in the **Table 14**.

Table 14

Interrupt Source	Starting Address
External Request 0	03 (03_H)
Internal Timer/Counter 0	11 ($0B_H$)
External Request 1	19 (13_H)
Internal Timer/Counter 1	27 ($1B_H$)
Serial Interface	35 (23_H)
Teletext Sync Signals	43 ($2B_H$)
Analog Digital Converter	51 (33_H)

6.3.4 Interrupt Control

The information flags, which control the entire interrupt system, are stored in following special function registers:

IE	Interrupt Enable Register	$A8_H$
IP0	Interrupt Priority Register 1	$A9_H$
IP1	Interrupt Priority Register 2	AA_H
IRCON	Interrupt Control	AB_H
TCON	Timer/Counter Control Register	88_H
SCON	Serial Control Register	98_H
TTXSIR	Sync Interrupt Request Register	$C8_H$
ACQSIR	Acquisition Sync Interrupt Register	$C0_H$
ADCON	ADC-Control Register	$D8_H$

The interrupt system is shown diagrammatically in **Figure 23**.

A source requests an interrupt by setting its associated interrupt request flag in the TCON, SCON, TTXSIR, ACQSIR or ADCON- register, as described in detail in **Table 15**.

Table 15

Interrupt Source	Request Flag	Bit Location
External Request 0	IE0	TCON.1
Internal Timer/Counter 0	TF0	TCON.5
External Request 1	IE1	TCON.3
Internal Timer/Counter 1	TF1	TCON.7
Serial Interface	RI/TI	SCON.0/.1
Teletext Sync Signals	DVIRST	TTXSIR.2
	DHIRST	TTXSIR.0
	EVENST	ACQSIR.6
	LIN24ST	ACQSIR.4
	AVIRST	ACQSIR.2
	AHIRST	ACQSIR.0
Analog Digital Converter	IADC	ADCON.5

The timer 0 and timer 1 interrupts are generated by TF0 and TF1, which are set by a rollover in their respective timer/counter register, except for timer 0 in mode 3.

The serial interface interrupt (receive or transmit) is generated when flag RI or TI is set. RI or TI will be set, when a byte has been received or transmitted over the serial port. For details see **Chapter “More about Mode 0” on page 95**, **Chapter “More about Mode 1” on page 95** and **Chapter “More about Modes 2 and 3” on page 96**.

The teletext sync signal interrupt is generated by setting and enabling at least one of six possible signal sources: two signals from the display clock system (V and H) and 4 signals from the acquisition clock system (start of even field, start of ACQ- line 24 in each field, V and H) as shown in **Figure 22**. The teletext sync signal interrupt is synchronous to the respective acquisition or display clock system. Thus clock synchronous software timers can be realized.

The analog digital converter interrupt is generated on completion of the analog digital conversion.

Within the IE-register there are eight addressable flags. Seven flags enable/disable the seven interrupt sources when set/cleared. Setting/clearing the eighth flag permits a global enable/disable of all enabled interrupt requests.

All the bits that generate interrupts can be set or cleared by software, with the same result as though they had been set or cleared by hardware. That is, interrupts can be generated or pending interrupt requests can be cancelled by software.

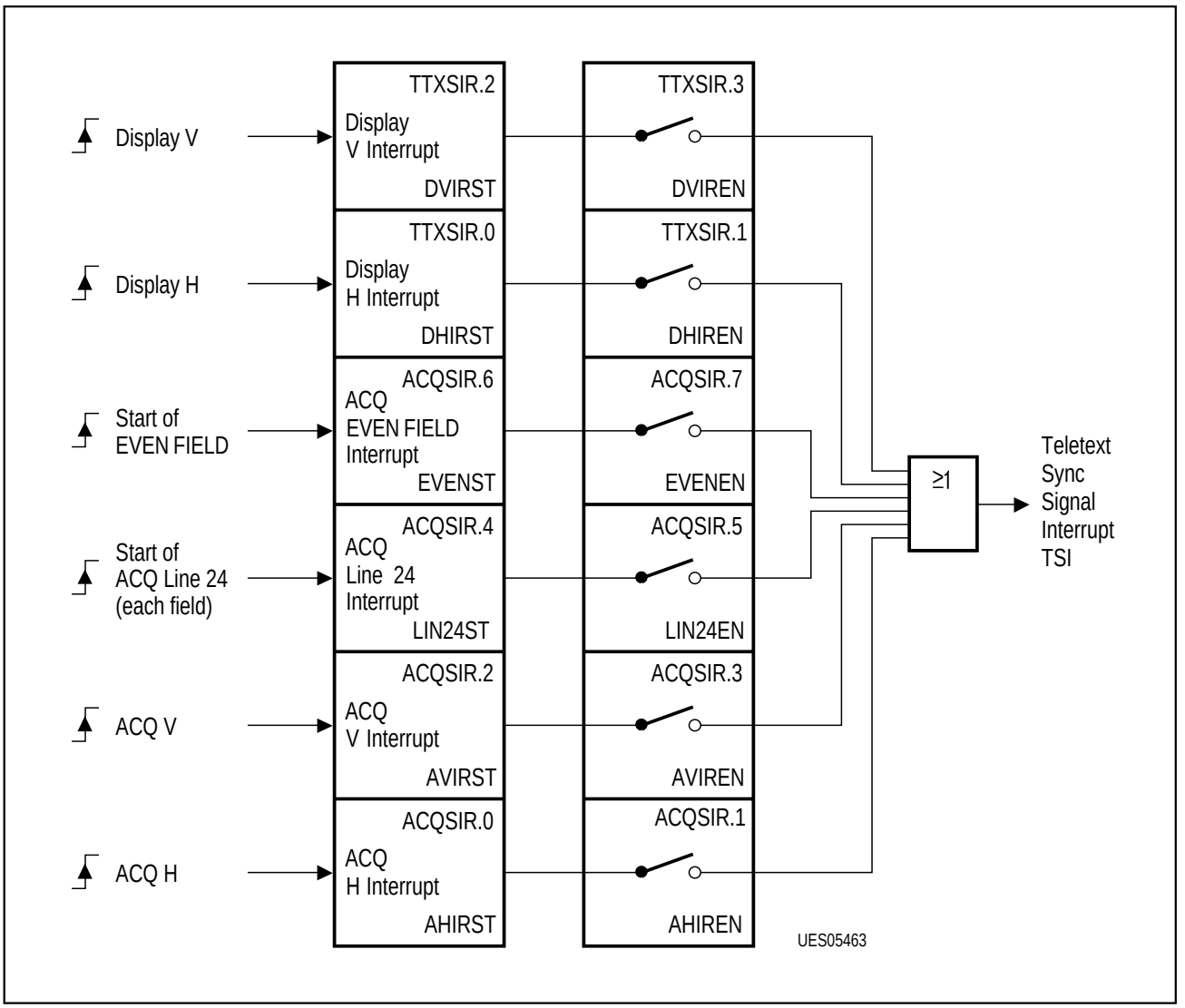
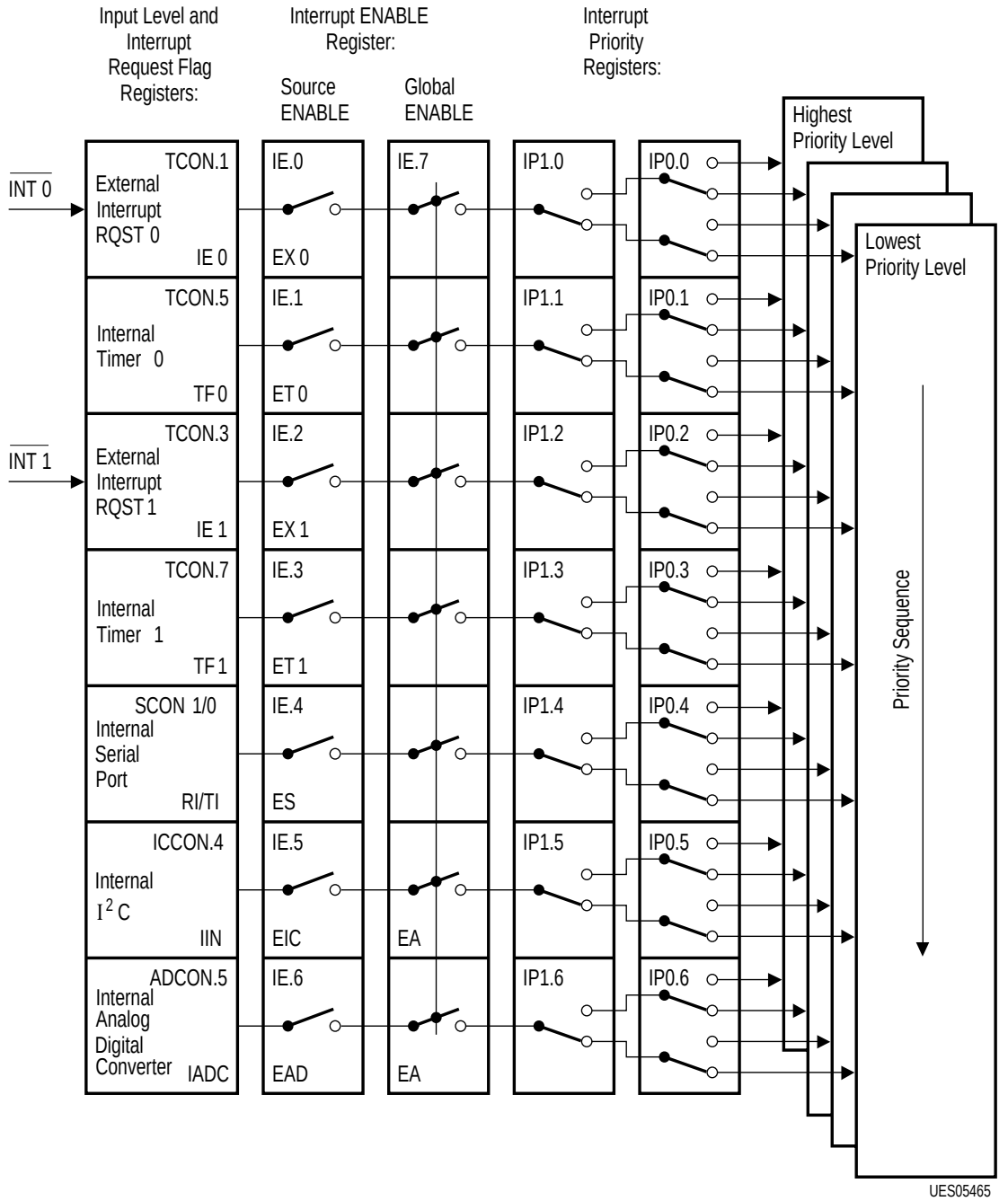


Figure 22
Teletext Sync Signal Interrupt System



- SEVEN INTERRUPT SOURCES
- EACH INTERRUPT CAN BE INDIVIDUALLY ENABLED/DISABLED
- ENABLED INTERRUPTS CAN BE GLOBALLY ENABLED/DISABLED
- EACH INTERRUPT CAN BE ASSIGNED TO EITHER OF FOUR PRIORITY LEVELS
- EACH INTERRUPT VECTORS TO A SEPARATE LOCATION IN PROGRAM MEMORY
- INTERRUPT NESTING TO FOUR LEVELS
- EXTERNAL INTERRUPT REQUESTS CAN BE PROGRAMMED TO BE LEVEL- OR TRANSITION-ACTIVATED

Figure 23
Interrupt System

Teletext Sync Interrupt Request Register TTXSIR

Teletext Sync Interrupt Request Register **TTXSIR** **SFR-Address $C8_H$**

Default after reset: 00_H

(MSB) (LSB)

–	VSY	HSY	PCLK	DVIREN	DVIRST	DHIREN	DHIRST

VSY, HSY, PCLK These bits are no interrupt bits. They are described in **Chapter “Display Special Function Registers” on page 24**.

DVIREN Enables or disables the display vertical sync interrupt request. If DVIREN = 1, this interrupt will be enabled.

DVIRST Display vertical sync interrupt request flag. Set by the rising edge of the display vertical sync pulse. Must be cleared by software.

DHIREN Enables or disables the display horizontal sync interrupt request.
If DHIREN = 1, this interrupt will be enabled.

DHIRST Display horizontal sync interrupt request flag. Set by the rising edge of the display horizontal sync pulse. Must be cleared by software.

Acquisition Sync Interrupt Request Register ACQSIR

Acquisition Sync Interrupt Request Register **ACQSIR** **SFR-Address $C0_H$**

Default after reset: 00_H

(MSB) (LSB)

EVENEN	EVENST	LIN24EN	LIN24ST	AVIREN	AVIRST	AHIREN	AHIRST

EVENEN Enables or disables the even field interrupt request.
If EVENEN = 1, this interrupt will be enabled.

EVENST Even field interrupt request flag. Set at the start of even field (field 1).
Must be cleared by software.

LIN24EN Enables or disables the acquisition line 24 interrupt request.
If LIN24EN = 1, this interrupt will be enabled.

LIN24ST Acquisition line 24 interrupt request flag. Set at the start of acquisition line 24 in each field. Must be cleared by software.

AVIREN Enables or disables the acquisition vertical sync interrupt request.
If AVIREN = 1, this interrupt will be enabled.

AVIRST Acquisition vertical sync interrupt request flag. Set by the rising edge of the acquisition vertical sync pulse. Must be cleared by software.

AHIREN Enables or disables the acquisition horizontal sync interrupt request.
If AHIREN = 1, this interrupt will be enabled.

AHIRST Acquisition horizontal sync interrupt request flag. Set by the rising edge of the acquisition horizontal sync pulse. Must be cleared by software.

Interrupt Enable Register IE

Interrupt Enable Register **IE** **SFR-Address $A8_H$**

Default after reset: 00_H

(MSB) (LSB)

EA	EADC	ETSI	ES	ET1	EX1	ET0	EX0

EA Enables or disables all interrupts. If EA = 0, no interrupt will be acknowledged.
If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.

EADC Enables or disables the analog digital converter interrupt. If EADC = 1, this interrupt will be enabled.

ETSI Enables or disables the teletext sync interrupts. If ETSI = 1, this interrupt will be enabled.

ES Enables or disables the serial interface interrupt. If ES = 1, this interrupt will be enabled.

ET1 Enables or disables the timer 1 overflow interrupt. If ET1 = 1, the timer 1 interrupt will be enabled.

EX1 Enables or disables external interrupt 1. If EX1 = 1, external interrupt 1 will be enabled.

ET0 Enables or disables the timer 0 overflow interrupt. If ET0 = 1, the timer 0 interrupt will be enabled.

EX0 Enables or disables external interrupt 0. If EX0 = 1, external interrupt 0 will be enabled.

Interrupt Priority Register IP0 and IP1

Interrupt Priority Register IP0 SFR-Address $A9_H$

Default after reset: 00_H

(MSB) (LSB)

–	IP0.6	IP0.5	IP0.4	IP0.3	IP0.2	IP0.1	IP0.0

Interrupt Priority Register IP1 SFR-Address AA_H

Default after reset: 00_H

(MSB) (LSB)

–	IP1.6	IP1.5	IP1.4	IP1.3	IP1.2	IP1.1	IP1.0

Corresponding bit-locations in both registers are used to set the interrupt priority level of an interrupt.

Table 16

IP1.X	IP0.X	Function
0	0	Set priority level 0 (lowest)
0	1	Set priority level 1
1	0	Set priority level 2
1	1	Set priority level 3 (highest)

Table 17

Bit	Corresponding Interrupt
IP1.0 / IP0.0	IE0
IP1.1 / IP0.1	TF0
IP1.2 / IP0.2	IE1
IP1.3 / IP0.3	TF1
IP1.4 / IP0.4	RI/TI
IP1.5 / IP0.5	DVIRST/ DHIRST/ EVENST/ LIN24ST/AVIRST/AHIRST
IP1.6 / IP0.6	IADC

Setting/clearing a bit in the IP-registers establishes its associated interrupt request priority level. If a low-priority level interrupt is being serviced, a higher-priority level interrupt will interrupt it. However, an interrupt source cannot interrupt a service program of the same or higher priority level.

If two requests of different priority levels are received simultaneously, the request of higher priority level will be serviced. If requests of the same priority level are received simultaneously, an internal polling sequence determines which request is serviced. Thus within each priority level there is a second priority structure determined by the polling sequence, see in **Table 18**.

Table 18

Source	Priority within Level
1. IE0	(highest)
2. TF0	
3. IE1	
4. TF1	
5. RI/TI	
6. DVIRST DHIRST EVENST LIN24ST AVIRST AHIRST	
7. IADC	(lowest)

Note that the “priority within level” structure is only used to resolve simultaneous requests of the same priority level.

6.3.4.1 Interrupt Nesting

The process whereby a higher-level interrupt request interrupts a lower-level interrupt service program is called nesting. In this case the address of the next instruction in the lower-priority service program is pushed onto the stack, the stack pointer is incremented by two and processor control is transferred to the program memory location of the first instruction of the higher-level service program. The last instruction of the higher-priority interrupt service program must be a RETI-instruction. This instruction clears the higher “priority-level-active” flip-flop. RETI also returns processor control to the next instruction of the lower-level interrupt service program. Since the lower “priority-level-active” flip-flop has remained set, higher priority interrupts are re-enabled while further lower-priority interrupts remain disabled.

6.3.4.2 External Interrupts

The external interrupt request inputs ($\overline{INT0}$ and $\overline{INT1}$) can be programmed for either transition- activated or level-activated operation. Control of the external interrupts is provided by the four low- order bits of TCON as shown in the follow section.

When IT0 and IT1 are set to one, interrupt requests on $\overline{INT0}$ and $\overline{INT1}$ are transition-activated (high-to-low), else they are low-level activated. IE0 and IE1 are the interrupt request flags. These flags are set when their corresponding interrupt request inputs at $\overline{INT0}$ and $\overline{INT1}$, respectively, are low when sampled by the processor and the transition-activated scheme is selected by IT0 and IT1.

Function of Lower Nibble Bits in TCON

Timer and Interrupt Control Register **TCON** **SFR-Address 88_H**

Default after reset: 00_H

(MSB) (LSB)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TCON.4 – TCON.7 See **Chapter “General Purpose Timers/Counters” on page 80**.

IE1 Interrupt 1 edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.

IT1 Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts. IT1 = 1 selects transition-activated external interrupts.

IE0 Interrupt 0 edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.

IT0 Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts. IT0 = 1 selects transition-activated external interrupts.

Interrupt Control Register **IRCON** **SFR-Address AB_H**

Default after reset: $XXXX0101_B$

(MSB)							(LSB)
–	–	–	–	**EX1R**	**EX1F**	**EX0R**	**EX0F**

EX1R if set, EX1-interrupt detection on rising edge at Pin P3.3

EX1F if set, EX1-interrupt detection on falling edge at Pin P3.3

EX0R if set, EX0-interrupt detection on rising edge at Pin P3.2

EX0F if set, EX0-interrupt detection on falling edge at Pin P3.2

– Transition-Activated Interrupts
(IT0 = 1, IT1 = 1)

The IE0, IE1 flags are set by a transition at $\overline{INT0}$, $\overline{INT1}$, respectively; they are cleared during entering the corresponding interrupt service routine.

For transition-activated operation, the input must remain active for more than six oscillator periods, but needs not to be synchronous with the oscillator. The opposite transition of a transition-activated input may occur at any time after the six oscillator period latching time, but the input must remain inactive for six oscillator periods before reactivation.

– Level-Activated Interrupts
(IT0 = 0, IT1 = 0)

The IE0, IE1 flags are set whenever $\overline{INT0}$, $\overline{INT1}$ are respectively sampled at low level. Sampling $\overline{INT0}$, $\overline{INT1}$ at high level clears IE0, IE1, respectively.

For level-activated operation, if the input is active during the sampling that occurs seven oscillator periods before the end of the instruction in progress, an interrupt subroutine call is made. The level-activated input needs to be low only during the sampling that occurs seven oscillator periods before the end of the instruction in progress and may remain low during the entire execution of the service program. However, the input must be deactivated before the service routine is completed to avoid invoking a second interrupt, or else another interrupt will be generated.

Extension of Standard 8051 Interrupt Logic

For more flexibility, the SDA 525x family provides a new feature in detection EX0 and EX1 in edge-triggered mode. Now there is the possibility to trigger an interrupt on the falling and / or rising edge at the dedicated Port3-Pin. Therefore, an additional register IRCON has been defined, which is described on the top.

6.3.4.3 Interrupt Task Function

The processor records the active priority level(s) by setting internal flip-flop(s). Each interrupt level has its own flip-flop. The flip-flop corresponding to the interrupt level being serviced is reset when the processor executes a RETI-instruction.

The sequence of events for an interrupt is:

– A source provokes an interrupt by setting its associated interrupt request bit to let the processor know an interrupt condition has occurred.
– The interrupt request is conditioned by bits in the interrupt enable and interrupt priority registers.
– The processor acknowledges the interrupt by setting one of the four internal “priority-level active” flip-flops and performing a hardware subroutine call. This call pushes the PC (but not the PSW) onto the stack and, for some sources, clears the interrupt request flag.
– The service program is executed.
– Control is returned to the main program when the RETI-instruction is executed. The RETI- instruction also clears one of the internal “priority-level active” flip-flops.

The interrupt request flags IE0, IE1, TF0 and TF1 are cleared when the processor transfers control to the first instruction of the interrupt service program. The RI/TI, DVIRST, DHIRST, EVENST, LIN24ST, AVIRST, AHIRST and IADC-interrupt request flags must be cleared as part of the respective interrupt service program.

6.3.4.4 Response Time

The highest-priority interrupt request gets serviced at the end of the instruction in progress unless the request is made in the last seven (CDC=0) oscillator periods of the instruction in progress. Under this circumstance, the next instruction will also execute before the interrupt's subroutine call is made.

If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The call itself takes two cycles. Thus, a minimum of three complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine. If the instruction in progress is not in its final cycle, the additional wait time cannot be more than 3 cycles, since the longest instructions (MUL and DIV) are only 4 cycles long, and if the instruction in progress is RETI or an access to IE or IP0 and IP1, the additional wait time cannot be more than 5 cycles (a maximum of one more cycle to complete the instruction in progress, plus 4 cycles to complete the next instruction if the instruction is MUL or DIV). Thus, in a single-interrupt system, the response time is always more than 3 cycles and less than 8 cycles (approximately 2.33 µs at 18-MHz operation). Note, that a machine cycle can consist of 12 oscillator periods (CDC = 1) or only six oscillator periods (CDC = 0) (see **Chapter “Advanced Function Register” on page 115**). Examples of the best and worst case conditions are illustrated in **Table 19**.

Table 19

Instruction	Time (Oscillator Periods)			
	Best Case		Worst Case	
External interrupt generated immediately before (best) / after (worst) the pin is sampled (time until end of bus cycle).	2 + ε	[1 + ε]	2 – ε	[1 – ε]
Current or next instruction finishes in 12-[6-] oscillator periods	12	[6]	12	[6]
Next instruction is MUL or DIV	don't care		48	[24]
Internal latency for hardware subroutine call	24	[12]	24	[12]
	38	[19]	86	[43]

Note: values without brackets apply for CDC = 1 and values in brackets for CDC = 0 (see **Chapter "Advanced Function Register" on page 115**).

If an interrupt of equal or higher priority level is already in progress, the additional wait time obviously depends on the nature of the other interrupt's service routine.

6.3.5 Processor Reset and Initialization

Processor initialization is accomplished with activation of the $\overline{\text{RST}}$ pin, which is the input to a Schmitt Trigger. To reset the processor, this pin should be held low for at least four machine cycles, while the oscillator is running stable. Upon powering up, $\overline{\text{RST}}$ should be held low for at least 10 ms after the power supply stabilizes to allow the oscillator to stabilize. Crystal operation below 6 MHz will increase the time necessary to hold $\overline{\text{RST}}$ low. Two machine cycles after receiving of $\overline{\text{RST}}$, the processor ceases from instruction execution and remains dormant for the duration of the pulse. The high-going transition then initiates a sequence which requires approximately one machine cycle to execute before normal operation commences with the instruction at absolute location 0000_H. Program memory locations 0000_H through 0002_H are reserved for the initialization routine of the microcomputer. This sequence ends with registers initialized as shown in **Chapter "Memory Organization" on page 49**.

After the processor is reset, all ports are written with one (1). Outputs are undefined until the reset period is complete.

An automatic reset can be obtained when V_{DD} is turned on by connecting the $\overline{\text{RST}}$-pin to V_{SS} through a 10 µF capacitor, providing the V_{DD} rise time does not exceed a millisecond and the oscillator start-up time does not exceed 10 milliseconds. When power comes on, the current drawn by $\overline{\text{RST}}$-pin starts to charge the capacitor. The voltage V_{RST} at $\overline{\text{RST}}$-pin is the capacitor voltage, and increases to V_{DD} as the capacitor charges. The larger the

capacitor, the more slowly V_{RST} decreases. V_{RST} must remain below the lower threshold of the Schmitt Trigger long enough to effect a complete reset. The time required is the oscillator start-up time plus 4 machine cycles.

Attention: While reset is active and at least four machine cycles after rising edge of $\overline{RST}$, ALE, P4.0 and P3.6 should not be pulled down externally. Otherwise a special production test mode is entered.

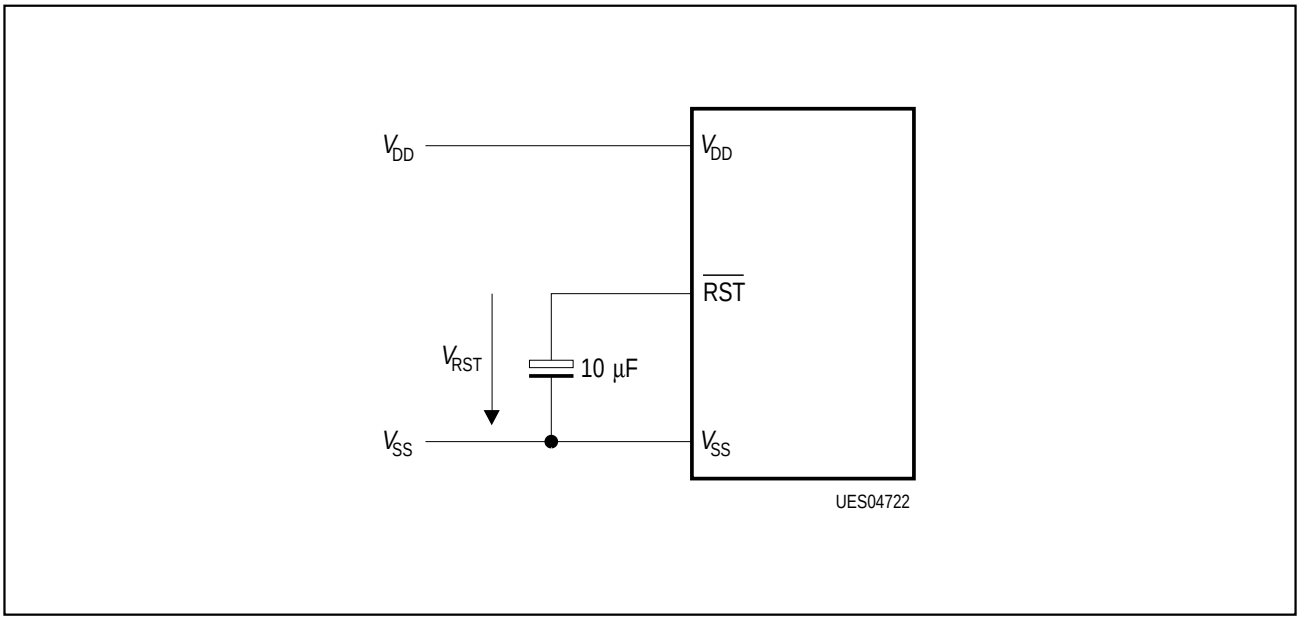


Figure 24
Power-On Reset Circuit

Power-Down Operations

The controller provides two modes in which power consumption can be significantly reduced.

– Idle mode. The CPU is gated off from the oscillator. All peripherals are still provided with the clock and are able to work.
– Power-down mode. Operation of the controller is turned off. This mode is used to save the contents of internal RAM with a very low standby current.

Both modes are entered by software. Special function register PCON is used to enter one of these modes.

Power Control Register PCON

Power Control Register **PCON** **SFR-Address 87_H**

Default after reset: 000xxx00

(MSB) (LSB)

SMOD	PDS	IDLS	–	–	–	PDE	IDLE

PDS Power-down start bit. The instruction that sets the PDS-flag is the last instruction before entering the power down mode.

IDLS IDLE start bit. The instruction that sets the PDS-flag is the last instruction before entering the idle mode.

PDE Power-down enable bit. When set, starting the power-down mode is enabled.

IDLE Idle enable bit. When set, starting the idle mode is enabled.

SMOD Baud rate control for serial interface; if set, the baud rate is doubled.

Entering the idle mode is done by two consecutive instructions immediately following each other. The first instruction has to set bit IDLE (PCON.0) and must not set bit IDLS (PCON.5). The following instruction has to set bit IDLS (PCON.5) and must not set bit IDLE (PCON.0). Bits IDLE and IDLS will automatically be cleared after having been set. This double-instruction sequence is implemented to minimize the chance of unintentionally entering the idle mode. The following instruction sequence may serve as an example:

ORL PCON,#00000001_B Set bit IDLE, bit IDLS must not be set.

ORL PCON,#00100000_B Set bit IDLS, bit IDLE must not be set.

The instruction that sets bit IDLS is the last instruction executed before going into idle mode.

The idle mode can be terminated by activation of any enabled interrupt (or a hardware reset). The CPU-operation is resumed, the interrupt will be serviced and the next instruction to be executed after RETI-instruction will be the one following the instruction that set the bit IDLS. The port state and the contents of SFRs are held during idle mode.

Entering the power-down mode is done by two consecutive instructions immediately following each other. The first instruction has to set bit PDE (PCON.1) and must not set bit PDS (PCON.6). The following instruction has to set bit PDS (PCON.6) and must not set bit PDE (PCON.1). Bits PDE and PDS will automatically be cleared after having been set. This double-instruction sequence is implemented to minimize the chance of

unintentionally entering the power-down mode. The following instruction sequence may serve as an example:

ORL PCON,#00000010_B Set bit PDE, bit PDS must not be set.

ORL PCON,#01000000_B Set bit PDS, bit PDE must not be set.

The instruction that sets bit PDS is the last instruction executed before going into power-down mode.

If idle mode and power-down mode are invoked simultaneously, the power-down mode takes precedence.

The only exit from power-down mode is a hardware reset. The reset will redefine all SFRs, but will not change the contents of internal RAM.

6.3.6 Ports and I/O-Pins

There are 26 I/O-pins configured as three 8-bit ports, one 4-bit-port (P2.0 – 2.3) and one 2-bit port (P4.0 – 4.1, P4.1 for ROM-less version only). Each pin can be individually and independently programmed as input or output and each can be configured dynamically.

An instruction that uses a port's bit/byte as a source operand reads a value that is the logical AND of the last value written to the bit/byte and the polarity being applied to the pin/pins by an external device (this assumes that none of the processor's electrical specifications are being violated). An instruction that reads a bit/byte, operates on the content, and writes the result back to the bit/byte, reads the last value written to the bit/byte instead of the logic level at the pin/pins. Pins comprising a single port can be made a mixed collection of inputs and outputs by writing a “one” to each pin that is to be an input. Each time an instruction uses a port as the destination, the operation must write “ones” to those bits that correspond to the input pins. An input to a port pin needs not to be synchronized to the oscillator.

All the port latches have “one” s written to them by the reset function. If a “zero” is subsequently written to a port latch, it can be reconfigured as an input by writing a “one” to it.

The instructions that perform a read of, operation on, and write to a port's bit/byte are INC, DEC, CPL, JBC, SETB, CLR, MOV P.X, CJNE, DJNZ, ANL, ORL, and XRL. The source read by these operations is the last value that was written to the port, without regard to the levels being applied at the pins. This insures that bits written to a “one” (for use as inputs) are not inadvertently cleared.

Port 0 has an open-drain output. Writing a “one” to the bit latch leaves the output transistor off, so the pin floats.

In that condition it can be used as a high-impedance input. Port 0 is considered “true bidirectional”, because when configured as an input it floats.

Ports 1, 3 and 4 have “quasi-bidirectional” output drivers which comprise an internal pullup resistor of 10 kΩ to 40 kΩ. When configured as inputs they pull high and will

source current when externally pulled low (for details see **Chapter “DC-Characteristics” on page 129**).

In ports P1, P3 and P4 the output drivers provide source current for one oscillator period if, and only if, software updates the bit in the output latch from a “zero” to an “one”. Sourcing current only on “zero to one” transition prevents a pin, programmed as an input, from sourcing current into the external device that is driving the input pin.

Secondary functions can be selected individually and independently for the pins of port 1 and 3. Further information on port 1's secondary functions is given in **Chapter “Pulse Width Modulation Unit (PWM)” on page 106**. P3 generates the secondary control signals automatically as long as the pin corresponding to the appropriate signal is programmed as an input, i. e. if the corresponding bit latch in the P3 special function register contains a “one”.

The following alternate functions can be selected when using the corresponding P3 pins:

P3.0	ODD/EVEN	(ODD/EVEN-indicator output)
P3.2	$\overline{\text{INT0}}$	(external interrupt 0)
P3.3	$\overline{\text{INT1}}$	(external interrupt 1)
P3.4	T0	(Timer/Counter 0 external input)
P3.5	T1	(Timer/Counter 1 external input)
P3.6	RXD	(serial port receive line)
P3.7	TXD	(serial port transmit line)

Read Modify-Write Feature

“Read-modify-write” commands are instructions that read a value, possibly change it, and then rewrite it to the latch. When the destination operand is a port or a port bit, these instructions read the latch rather than the pin. The read-modify-write instructions are listed in **Table 20**.

The read-modify-write instructions are directed to the latch rather than the pin in order to avoid a possible misinterpretation of the voltage level at the pin. For example, a port bit might be used to drive the base of a transistor. When a “one” is written to the bit, the transistor is turned on.

If the CPU then reads the same port bit at the pin rather than the latch, it will read the base voltage of the transistor and interpret it as a 0. Reading the latch rather than the pin will return the correct value of “one”.

Table 20
Read-Modify-Write Instructions

Mnemonic	Description	Example
ANL	logical AND	ANL P1, A
ORL	logical OR	ORL P2, A
XRL	logical EX – OR	XRL P3, A
JBC	jump if bit = 1 and clear bit	JBC P1.1, LABEL
CPL	complement bit	CPL P3.0
INC	increment	INC P1
DEC	decrement	DEC P1
DJNZ	decrement and jump if not zero	DJNZ P3, LABEL
MOV PX.Y, C(1)	move carry bit to bit Y of Port X	MOV P1.7, C
CLR PX.Y(1)	clear bit Y of Port X	CLR P2.6
SET PX.Y(1)	set bit Y of Port X	SET P3.5

(1) The instruction reads the port byte (all 8 bits), modifies the addressed bit, then writes the new byte back to the latch

6.3.7 General Purpose Timers/Counters

Two independent general purpose 16-bit timers/ counters are integrated for use in measuring time intervals, measuring pulse widths, counting events, and causing periodic (repetitive) interrupts. Either can be configured to operate as timer or event counter.

In the “timer” function, the registers TLx and/or THx (x = 0, 1) are incremented once every machine cycle. Thus, one can think of it as counting machine cycles.

A machine cycle consists of 6 or 12 oscillator periods. This depends on the setting of bit CDC in the Advanced Function Register AFR of the special function registers (see **Chapter “Advanced Function Register” on page 115**). For CDC = 1 a machine cycle consists of 12 oscillator periods and for CDC = 0 of 6 oscillator periods.

In the “counter” function, the registers TLx and/or THx (x = 0, 1) are incremented in response to a 1-to-0 transition at its corresponding external input pin, T0 or T1. In this function, the external input is sampled during every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during the cycle following the one in which the transition was detected. Since it takes 2 machine cycles (24 oscillator periods for CDC = 1 or 12 oscillator periods for CDC = 0) to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency for CDC = 1 or 1/12 of the oscillator frequency for CDC = 0. There are no restrictions on the duty cycle of the external input signal, but to ensure that a given level is sampled at least once before it changes, it should be held for at least one full machine cycle.

Timer/Counter 0: Mode Selection

Timer/counter 0 can be configured in one of four operating modes, which are selected by bit-pairs (M1, M0) in TMOD-register (see **page 83**).

– Mode 0

Putting timer/counter 0 into mode 0 makes it look like an 8048 timer, which is an 8-bit counter with a divide-by-32 prescaler. **Figure 25** shows the mode 0 operation as it applies to timer 0.

In this mode, the timer register is configured as a 13-bit register. As the count rolls over from all 1 s to all 0 s, it sets the timer interrupt flag TF0. The counted input is enabled to the timer when TR0 = 1 and either GATE = 0 or $\overline{INT0}$ = 1. (Setting GATE = 1 allows the timer to be controlled by external input $\overline{INT0}$, to facilitate pulse width measurements.) TR0 is a control bit in the special function register TCON (see **page 84**). GATE is contained in register TMOD (see **page 83**).

The 13-bit register consists of all 8 bits of TH0 and the lower 5 bits of TL0. The upper 3 bits of TL0 are indeterminate and should be ignored. Setting the run flag (TR0) does not clear the registers.

– Mode 1

Mode 1 is the same as mode 0, except that the timer/counter 0 register is being run with all 16 bits.

– Mode 2

Mode 2 configures the timer/counter 0 register as an 8-bit counter (TL0) with automatic reload, as shown on see **page 83**. Overflow from TL0 not only sets TF0, but also reloads TL0 with the contents of TH0, which is preset by software. The reload leaves TH0 unchanged.

– Mode 3

Timer/counter 0 in mode 3 establishes TL0 and TH0 as two separate counters. The logic for mode 3 on timer 0 is shown in **Figure 27**. TL0 uses the timer 0 control bits: C/T, GATE, TR0, $\overline{INT0}$ and TF0. TH0 is locked into a timer function (counting machine cycles) and takes over the use of TR1 and TF1 from timer 1. Thus, TH0 now controls the “timer 1” interrupt.

Mode 3 is provided for applications requiring an extra 8-bit timer or counter. With timer 0 in mode 3, the processor can operate as if it has three timers/counters. When timer 0 is in mode 3, timer 1 can be turned on and off by switching it out of and into its own mode 3, or can still be used in any application not requiring an interrupt.

Timer/Counter 1: Mode Selection

Timer/counter 1 can also be configured in one of four modes, which are selected by its own bitpairs (M1, M0) in TMOD-register.

The serial port receives a pulse each time that timer/counter 1 overflows. This pulse rate is divided to generate the transmission rate of the serial port.

Modes 0 and 1 are the same as for counter 0.

– Mode 2

The “reload” mode is reserved to determine the frequency of the serial clock signal (not implemented).

– Mode 3

When counter 1's mode is reprogrammed to mode 3 (from mode 0, 1 or 2), it disables the increment counter. This mode is provided as an alternative to using the TR1 bit (in TCON-register) to start and stop timer/counter 1.

Configuring the Timer/Counter Input

The use of the timer/counter is determined by two 8-bit registers, TMOD (timer mode) and TCON (timer control), as shown on **page 83** and **84**. The input to the counter circuitry is from an external reference (for use as a counter), or from the on-chip oscillator (for use as a timer), depending on whether TMOD's C/T-bit is set or cleared, respectively. When used as a time base, the on-chip oscillator frequency is divided by twelve or six (**see Figure 25, 26** and **26**) before being used as the counter input. When TMOD's GATE bit is set (1), the external reference input (T1, T0) or the oscillator input is gated to the counter conditional upon a second external input ($\overline{\text{INT0}}$), ($\overline{\text{INT1}}$) being high. When the GATE bit is zero (0), the external reference, or oscillator input, is unconditionally enabled. In either case, the normal interrupt function of $\overline{\text{INT0}}$ and $\overline{\text{INT1}}$ is not affected by the counter's operation. If enabled, an interrupt will occur when the input at $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$ is low. The counters are enabled for incrementing when TCON's TR1 and TR0 bits are set. When the counters overflow, the TF1 and TF0 bits in TCON get set, and interrupt requests are generated.

The counter circuitry counts up to all 1's and then overflows to either 0's or the reload value. Upon overflow, TF1 or TF0 is set. When an instruction changes the timer's mode or alters its control bits, the actual change occurs at the end of the instruction's execution.

The T1 and T0 inputs are sampled near the falling-edge of ALE in the tenth, twenty-second, thirty-fourth and forty-sixth oscillator periods of the instruction-in-progress (CDC=1). Thus, an external reference's high and low times must each have a minimum duration of twelve oscillator periods for CDC = 1 or six oscillator periods for CDC = 0. There is a twelve (CDC = 1) or six (CDC = 0) oscillator period delay from the time when a toggled input (transition from high to low) is sampled to the time when the counter is incremented.

Timer/Counter Mode Register

Timer 0/1 Mode Register **TMOD** **SFR-Address 89_H**

Default after reset: 00_H

(MSB) (LSB)

GATE	C/T	M1	M0	GATE	C/T	M1	M0

Timer 1 Timer 0

GATE Gating control when set. Timer/counter “x” is enabled only while “$\overline{\text{INT}}$x” pin is high and “TRx” control bit is set. When cleared, timer “x” is enabled, whenever “TRx” control bit is set.

C/T Timer or counter selector. Cleared for timer operation (input from internal system clock). Set for Counter operation (input from “Tx” input pin).

Table 21

M1	M0	Operating Mode
0	0	SAB 8048 timer: “TLx” serves as five-bit prescaler.
0	1	16-bit timer/counter: “THx” and “TLx” are cascaded, there is no prescaler.
1	0	8-bit auto-reload timer/counter: “THx” holds a value which is to be reloaded into “TLx” each time it overflows.
1	1	(Timer 0) TL0 is an eight-bit timer/counter controlled by the standard timer 0 control bits; TH0 is an eight-bit timer only controlled by timer 1 control bits. (Timer 1) timer/counter 1 is stopped.

Timer/Counter Control Register

Timer 0/1 Mode Register **TCON** **SFR-Address 88_H**

Default after reset: 00_H

(MSB) (LSB)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1 Timer 1 overflow flag. Set by hardware on timer/counter overflow. Cleared by hardware when processor vectors to interrupt routine.

TR1 Timer 1 run control bit. Set/cleared by software to turn timer/counter on/off.

TF0 Timer 0 overflow flag. Set by hardware on timer/counter overflow. Cleared by hardware when processor vectors to interrupt routine.

TR0 Timer 0 run control bit. Set/cleared by software to turn timer/counter on/off.

IE1 Interrupt 1 edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.

IT1 Interrupt 1 type control bit. Set/cleared by software to specify edge/low level triggered external interrupts.

IE0 Interrupt 0 edge flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.

IT0 Interrupt 0 type control bit. Set/cleared by software to specify edge/low level triggered external interrupts.

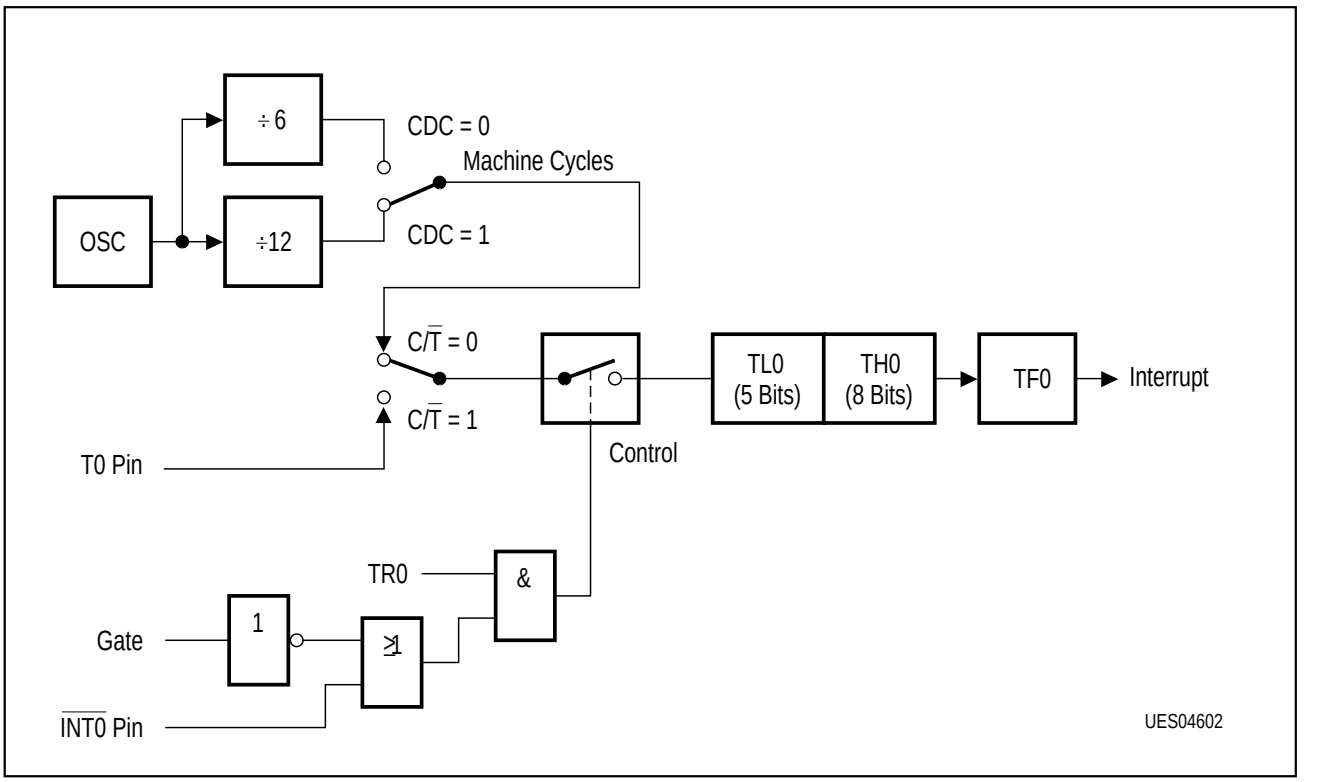


Figure 25
Timer/Counter 0 Mode 0: 13-Bit Counter

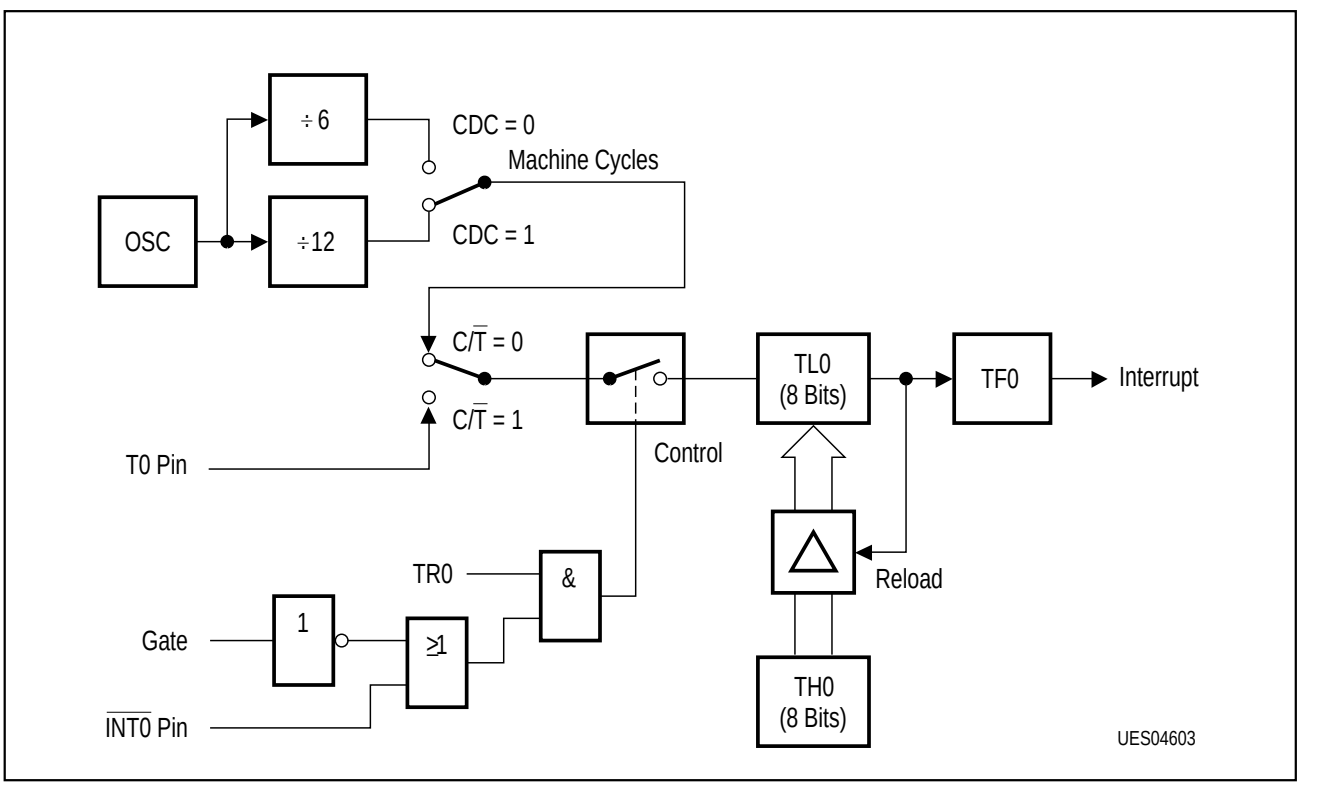


Figure 26
Timer/Counter 0 Mode 2: 8-Bit Auto-Reload

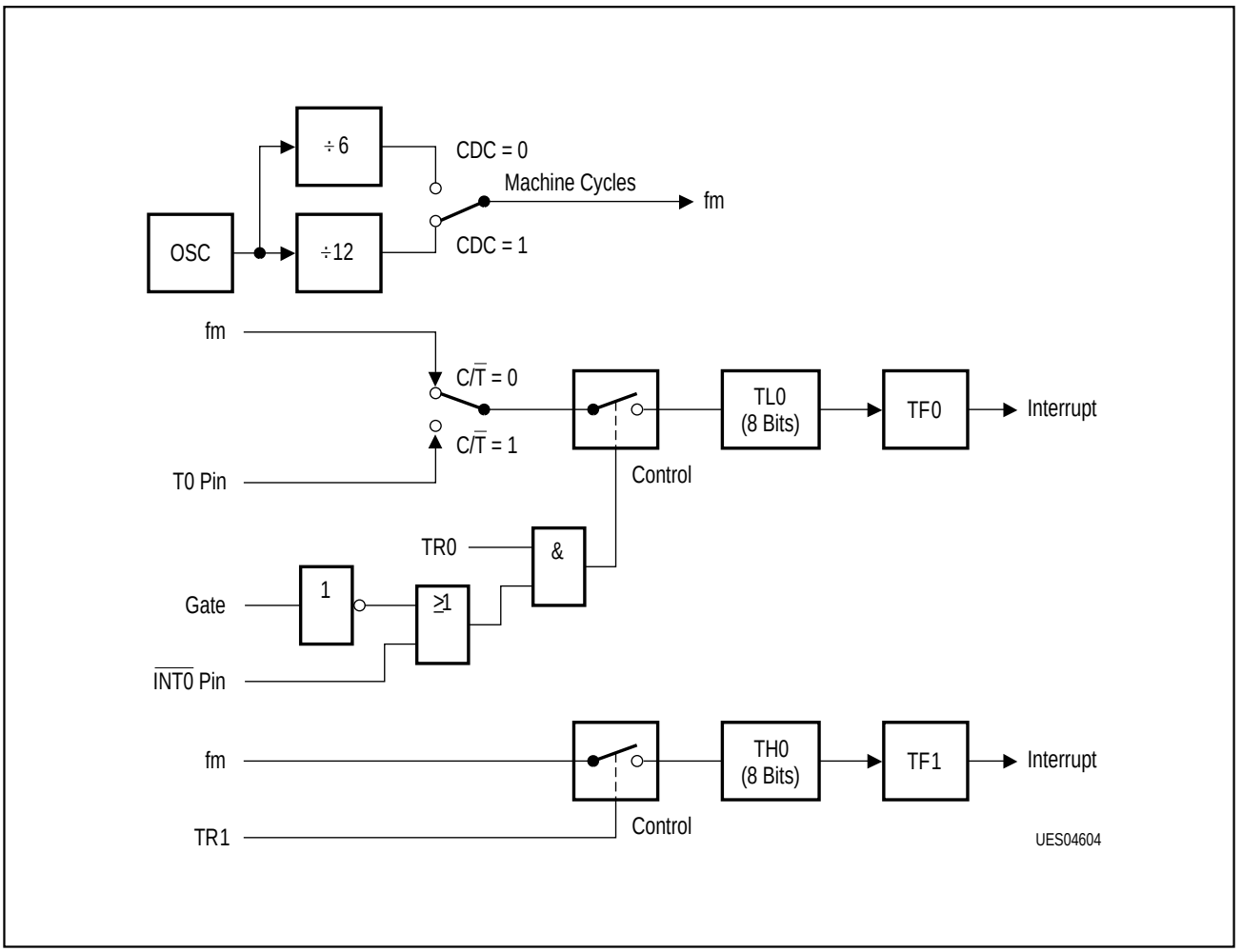


Figure 27
Timer/Counter 0 Mode 3: Two 8-Bit Counters

6.3.8 Watchdog Timer

To protect the systems against software upset, the user's program has to clear this watchdog within a previously programmed time period. If the software fails to do this periodical refresh of the watchdog timer, an internal hardware reset will be initiated. The software can be designed so that the watchdog times out if the program does not work properly.

The watchdog timer is a 15-bit timer, which is incremented by a count rate of either $f_{CYCLE}/2$ or $f_{CYCLE}/128$. The latter is enabled by setting bit WDTREL.7. Note, that f_{CYCLE} can be $f_{Quarz}/12$ for CDC = 1 or $f_{Quarz}/6$ for CDC = 0 (see **Chapter "Advanced Function Register" on page 115**). Immediately after start, the watchdog timer is initialized to the reload value programmed to WDTREL.0 – WDTREL.6. After an external reset register WDTREL is cleared to 00_H. The lower seven bits of WDTREL can be loaded by software at any time.

The watchdog timer is started by software by setting bit SWDT in special function register WDCON (bit 6). If the counter is stopped, and WDTREL is loaded with a new value, WDTH (high-byte of the watchdog timer) is updated immediately. WDTL (low-byte of the watchdog timer) is always zero, if the counter is stopped. Once started the watchdog timer cannot be stopped by software but can only be refreshed to the reload value by first setting bit WDT (AFR.6) and by the next instruction setting SWDT (WDCON.6). Bit WDT will automatically be cleared during the third machine cycle after having been set. This double instruction refresh of the watchdog timer is implemented to minimize the chance of an unintentional reset of the watchdog.

If the software fails to clear the watchdog in time, an internally generated watchdog reset is entered at the counter state $7FFF_H$. The duration of the reset signal then depends on the prescaler selection. This internal reset differs from an external reset only in so far as the watchdog timer is not disabled and bit WDTS (WDCON.7) is set. Bit WDTS allows the software to examine from which source the reset was activated. The watchdog timer status flag can also be cleared by software.

With WDTREL = 80_H and an oscillator frequency of 18 MHz the maximum time period is about 0.7 s for CDC = 0 and about 1.4 s for CDC = 1.

Watchdog Timer Control Bits

Watchdog Timer Reload Register **WDTREL** **SFR-Address 86_H**

Default after reset: 00_H

(MSB)							(LSB)
WDTREL.7	**WDTREL.6**	**WDTREL.5**	**WDTREL.4**	**WDTREL.3**	**WDTREL.2**	**WDTREL.1**	**WDTREL.0**

WDTREL.7 Prescaler bit. When set, the watchdog is clocked through an additional divide-by-64 prescaler.

WDTREL.0 - WDTREL.6 Seven bit reload value for the high-byte of the watchdog timer. This value is loaded to the WDT when a refresh is triggered by a consecutive setting of bits WDT and SWDT.

Watchdog Timer Control Register **WDCON** **SFR-Address $A7_H$**

Default after reset: 00_H

(MSB)							(LSB)
WDTS	**SWDT**	**–**	**–**	**–**	**–**	**–**	**–**

WDTS Watchdog timer reset flag. If bit WDTS is ‘1’ after reset, the reset has been initiated by the watchdog timer. After external reset, WDTS is reset to ‘0’.

SWDT Watchdog timer start flag. Set to activate the watchdog timer. When directly set after setting WDT, a watchdog timer refresh is performed.

WDCON.0 - WDCON.5 Reserved.

Advanced Function Register AFR

Advanced Function Register **AFR** **SFR-Address $A6_H$**

Default after reset: $00xxxxxx_B$

(MSB) (LSB)

CDC	**WDT**	**0**	**0**	**0**	**0**	**0**	**0**

CDC See **Chapter “Advanced Function Register” on page 115**.

WDT Watchdog timer refresh flag. Set to initiate a refresh of the watchdog timer. Must be set directly before SWDT (WDCON.6) is set to the watchdog timer.

AFR.0 - AFR.5 **Reserved, always to be written with ‘0’.**

6.3.9 Capture Compare Timer

For easier infrared signal decoding, an additional Capture Compare Timer is implemented. A functional overview is given in following feature list:

– 16-Bit-Counter with 2 or 3 prescaler bits selectable via SFR
– Counting rate: internal clock (18 MHz)
– Counter reloadable, prescaler bits reload with ‘0’
– Capture function
– Timer polling mode
– P3.3 or P3.2 selectable as capture input
– Capture on rising and/or falling edge
– Overflow-Bit

Infrared Timer Control Register IRTCON

Infrared Timer Control Register **IRTCON** **SFR-Address $E5_H$**

Default after reset: 00_H

(MSB) (LSB)

OV	PR	PLG	REL	RUN	RISE	FALL	SEL

OV will be set by hardware, if counter overflow has occurred; must be cleared by software

PR if cleared, 2-bit prescaler; if set, 3-bit prescaler

PLG if set, Timer polling mode selected, capture function is automatically disabled, reading capture registers will now show current timer value

REL if set, counter will be reloaded simultaneously with capture event

RUN run/stop the counter

RISE capture (and if REL=‘1’, reload) on rising edge

FALL capture (and if REL=‘1’, reload) on falling edge

SEL if set, P3.3 is selected for capture input, otherwise P3.2

Note: If counter is halted, a counter-reload with the contents of the reload registers is forced by hardware to give the counter a starting value.

The registers RELH and RELL (SFR-address $E2_H$ and $E1_H$) are the reload registers, CAPH and CAPL (SFR-addresses $E4_H$ and $E3_H$) are the corresponding capture registers. The reset value of these registers is undefined.

6.3.10 Serial Interface

The serial port is full duplex, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the receive register (however, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost). The serial port receive and transmit registers are both accessed at special function register SBUF. Writing to SBUF loads the transmit register, and reading SBUF accesses a physically separate receive register.

The frequencies and baud rates described in this chapter depend on the internal system clock, used by the serial interface. The internal system clock frequency of the serial interface is defined by the oscillator frequency f_{OSC} and the setting of bit CDC in the Advanced Function Register AFR of the special function registers (see **Chapter “Advanced Function Register” on page 115**).

The serial port can operate in 4 modes:

Mode 0: Serial data enters and exits through RxD (P3.6). TxD (P3.7) outputs the shift clock.

Mode 1: 10 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), and a stop bit (1). On reception, the stop bit goes into RB8 in special function register SCON. The baud rate is variable.

Mode 2: 11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmission, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On reception, the 9th data bit goes into RB8 in the special function register SCON, while the stop bit is ignored. The baud rate is programmable via SFR-Bit SMOD.

Mode 3: 11 bits are transmitted (through TxD) or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit and a stop bit (1). In fact, mode 3 is the same as mode 2 in all respects except the baud rate. The baud rate in mode 3 is variable.

Serial Port Control Register SCON

Serial Port Control Register **SCON** **SFR-Address 98_H**

Default after reset: 00_H

(MSB) (LSB)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0
SM1 Serial Port Mode Selection, see **Table 22**.

SM2 Enables the multiprocessor communication feature in modes 2 and 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0.

REN Enables serial reception. Set by software to enable reception. Cleared by software to disable reception.

TB8 Is the 9th data bit that will be transmitted in modes 2 and 3. Set or cleared by software as desired.

RB8 In modes 2 and 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.

TI Is the transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.

RI Is the receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through stop bit time in the other modes, in any serial reception. Must be cleared by software.

Table 22
Serial Port Mode Selection

SM0	SM1	Mode	Description	Baud Rate (CDC = 0)
0	0	0	Shift Reg.	$f_{OSC}/6$
0	1	1	8-bit UART	Variable
1	0	2	9-bit UART	$f_{OSC}/32, f_{OSC}/16$
1	1	3	9-bit UART	Variable

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1. The control, mode, and status bits of the serial port in special function register SCON are illustrated on **page 92**.

6.3.10.1 Multiprocessor Communication

Modes 2 and 3 of the serial interface of the controller have a special provision for multiprocessor communication. In these modes, 9 data bits are received. The 9th one goes into RB8. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB8 = 1. This feature is enabled by setting bit SM2 in SCON. A way to use this feature in multiprocessor communications is as follows.

When the master processor wants to transmit a block of data to one of the several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM2 = 1, no slave will be interrupted by a data byte. An address byte however, will interrupt all slaves, so that each slave can examine the received byte and see if it is being addressed. The addressed slave will clear its SM2 bit and prepare to receive the data bytes that will be coming. The slaves that weren't addressed leave their SM2s set and go on about their business, ignoring the coming data bytes.

SM2 has no effect in mode 0, and in mode 1 can be used to check the validity of the stop bit. In a mode 1 reception, if SM2 = 1, the receive interrupt will not be activated unless a valid stop bit is received.

6.3.10.2 Baud Rates

The baud rate in mode 0 is fixed:

$$\text{Mode 0 baud rate} = \frac{f_{OSC}}{6}$$

The baud rate in mode 2 depends on the value of bit SMOD in special function register PCON (bit 7). If SMOD = 0 (which is the value on reset), the baud rate is 1/32 of the oscillator frequency. If SMOD = 1, the baud rate is 1/16 of the oscillator frequency.

$$\text{Mode 2 baud rate} = \frac{2^{SMOD}}{32} \times f_{OSC}$$

The baud rates in modes 1 and 3 are determined by the timer 1 overflow rate or can be generated by the internal baud rate generator.

When timer 1 is used as the baud rate generator, the baud rates in modes 1 and 3 are determined by the timer 1 overflow rate and the value of SMOD as follows:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD}}{16} \times \text{Time 1 overflow}$$

The timer 1 interrupt should be disabled in this application. The timer itself can be configured for either “timer” or “counter” operation, and in any of the 3 running modes. In the most typical applications, it is configured for “timer” operation, in the auto-reload mode (high nibble of TMOD = 0010B). In that case, the baud rate is given by the formula:

$$\text{Mode 1, 3 baud rate} = \frac{2^{SMOD}}{16} \times \frac{f_{OSC}}{12 \times (256 - TH1)}$$

One can achieve very low baud rates with timer 1 by leaving the timer 1 interrupt enabled, configuring the timer to run as a 16-bit timer (high nibble of TMOD = 0001B), and using the timer 1 interrupt to do a 16-bit software reload.

6.3.10.3 More about Mode 0

Serial data enters and exits through RxD. TxD outputs the shift clock. 8 bits are transmitted/ received: 8 data bits (LSB first).

Figure 28 shows a simplified functional diagram of the serial port in mode 0, and associated timing.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write-to SBUF” signal also loads a 1 into the 9th bit position of the transmit shift register and tells the TX-control block to commence a transmission. The internal timing is such that one full machine cycle will elapse between “write-to-SBUF” and activation of SEND. SEND enables the output of the shift register to the alternate output function line of P3.6, and also enables SHIFT CLOCK to the alternate output function, line of P3.7. At the end of every machine cycle in which SEND is active, the contents of the transmit shift register is shifted one position to the right.

As data bits shift out to the right, zeros come in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position, is just left of the MSB, and all positions to the left of that contain zeros.

This condition flags the TX-control block to do one last shift and then deactivate SEND and set Tl. Both of these actions occur in the 10th machine cycle after “write-to-SBUF”.

Reception is initiated by the condition REN = 1 and Rl = 0. At the end of the next machine cycle, the RX-control unit writes the bits ‘1111 1110’ to the receive shift register, and the next clock phase activates RECEIVE.

RECEIVE enables SHIFT CLOCK to the alternate output function line of P3.7. At the end of every machine cycle in which RECEIVE is active, the contents of the Receive Shift register are shifted one position to the left. The value that comes in from the right is the value that was sampled at the P3.6 pin in the same machine cycle.

As data bits come in from the right, 1 s shift out to the left. When the 0 that was initially loaded into the rightmost position arrives at the leftmost position in the shift register, it flags the RX-control block to do one last shift and load SBUF. In the 10th machine cycle after the write to SCON that cleared Rl, RECEIVE is cleared and Rl is set.

6.3.10.4 More about Mode 1

Ten bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first) and a stop bit (1). On reception, the stop bit goes into RB8 in SCON.

The baud rate is determined by the timer 1 overflow rate.

Figure 30 shows a simplified functional diagram of the serial port in mode 1, and associated timings for transmit and receive.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write-to SBUF” signal also loads a ‘1’ into the 9th bit position of the transmit shift register and flags the TX- control block that a transmission is requested. Transmission actually

commences at the beginning of the machine cycle following the next rollover in the divide-by-16 counter (thus, the bit times are synchronized to the divide-by-16 counter, not to the “write-to-SBUF” signal).

The transmission begins with activation of $\overline{\text{SEND}}$, which puts the start bit to TxD. One bit time later, DATA is activated, which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that.

As data bits shift out to the right, zeros are clocked in from the left. When the MSB of the data byte is at the output position of the shift register, then the 1 that was initially loaded into the 9th position is just left of the MSB, and all positions to the left of that contain zeros. This condition flags the TX-control unit to do one last shift and then deactivate $\overline{\text{SEND}}$ and set TI. This occurs at the 10th divide-by-16 rollover after “write-to-SBUF”.

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and 1 FF_H is written into the input shift register. Resetting the divide-by-16 counter aligns its rollovers with the boundaries of the incoming bit times.

The 16 states of the counter divide each bit time into 16ths. At the 7th, 8th and 9th counter states of each bit time, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. This is done for noise rejection. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back looking for another 1-to-0 transition. This is to provide rejection of false start bits. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed.

As data bits come in from the right, 1 s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in mode 1 is a 9-bit register), it flags the RX-control block to do one last shift, load SBUF and RB8, and set RI. The signal to load SBUF and RB8, and to set RI, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated:

1. RI = 0, and
2. either SM2 = 0 or the received stop bit = 1

If either of these two conditions is not met, the received frame is irretrievably lost. If both conditions are met, the stop bit goes into RB8, the 8 data bits go into SBUF and RI is activated. At this time, no matter whether the above conditions are met or not, the unit goes back looking for a 1-to-0-transition in RxD.

6.3.10.5 More about Modes 2 and 3

11 bits are transmitted (through TxD), or received (through RxD): a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit, (1). On transmission, the 9th data bit (TB8) can be assigned the value of 0 or 1. On reception, the 9th data bit goes into RB8 in SCON.

Figures 32 and 34 show a functional diagram of the serial port in modes 2 and 3 and associated timings. The receive portion is exactly the same as in mode 1. The transmit portion differs from mode 1 only in the 9th bit of the transmit shift register.

Transmission is initiated by any instruction that uses SBUF as a destination register. The “write-to- SBUF” signal also loads TB8 into the 9th bit position of the transmit shift register and flags the TX- control unit that a transmission is requested. Transmission commences at the beginning of the machine cycle following the next rollover in the divide-by-16 counter (thus, the bit times are synchronized to the divide-by-16 counter, not to the “write-to-SBUF” signal).

The transmission begins with activation of $\overline{\text{SEND}}$, which puts the start bit to TxD. One bit time later, DATA is activated which enables the output bit of the transmit shift register to TxD. The first shift pulse occurs one bit time after that. The first shift clocks a 1 (the stop bit) into the 9th bit position of the shift register. Thereafter, only zeros are clocked in. Thus, as data bits shift out to the right, zeros are clocked in from the left. When TB8 is at the output position of the shift register, then the stop bit is just left of the TB8, and all positions to the left of that contain zeros.

This condition flags the TX-control unit to do one last shift and then deactivate $\overline{\text{SEND}}$ and set Tl. This occurs at the 11th divide-by-16 rollover after “write-to-SBUF”.

Reception is initiated by a detected 1-to-0 transition at RxD. For this purpose RxD is sampled at a rate of 16 times whatever baud rate has been established. When a transition is detected, the divide-by-16 counter is immediately reset, and $1FF_H$ is written to the input shift register.

At the 7th, 8th, and 9th counter states of each bit time, the bit detector samples the value of RxD. The value accepted is the value that was seen in at least 2 of the 3 samples. If the value accepted during the first bit time is not 0, the receive circuits are reset and the unit goes back looking for another 1-to-0 transition. If the start bit proves valid, it is shifted into the input shift register, and reception of the rest of the frame will proceed. As data bits come in from the right, 1 s shift out to the left. When the start bit arrives at the leftmost position in the shift register (which in modes 2 and 3 is a 9-bit register), it flags the RX-control block to do one last shift, load SBUF and RB8, and set Rl. The signal to load SBUF and RB8, and to set Rl, will be generated if, and only if, the following conditions are met at the time the final shift pulse is generated:

1. Rl = 0, and
2. either SM2 = 0 or the received 9th data bit = 1

If either of these two conditions is not met, the received frame is irretrievably lost, and Rl is not set. If both conditions are met, the received 9th data bit goes into RB8, the first 8 data bits go into SBUF. One bit time later, no matter whether the above conditions are met or not, the unit goes back looking for a 1-to-0-transition at the RxD input.

Note that the value of the received stop bit is irrelevant to SBUF, RB8 or Rl.

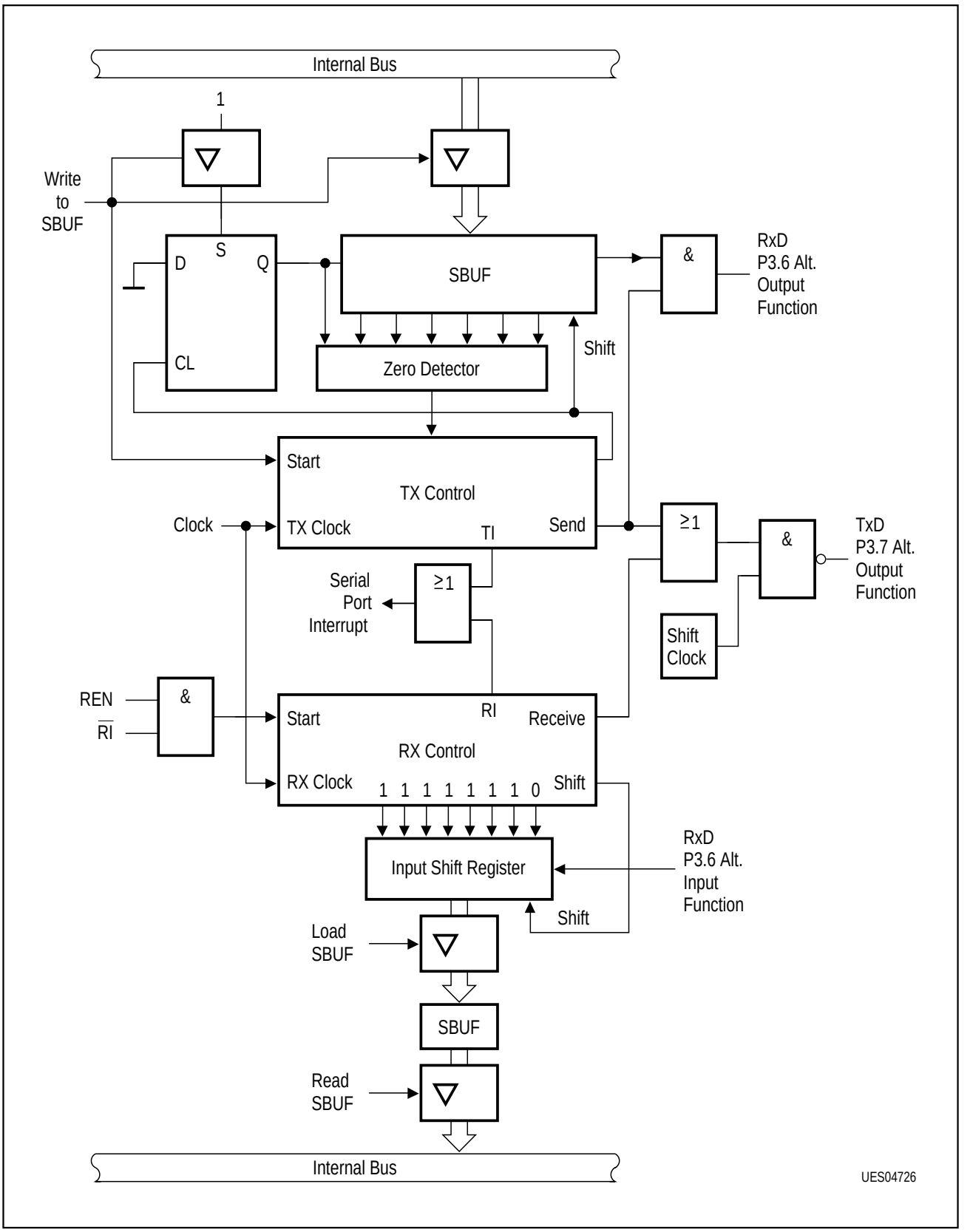


Figure 28
Serial Port Mode 0, Functional Diagram

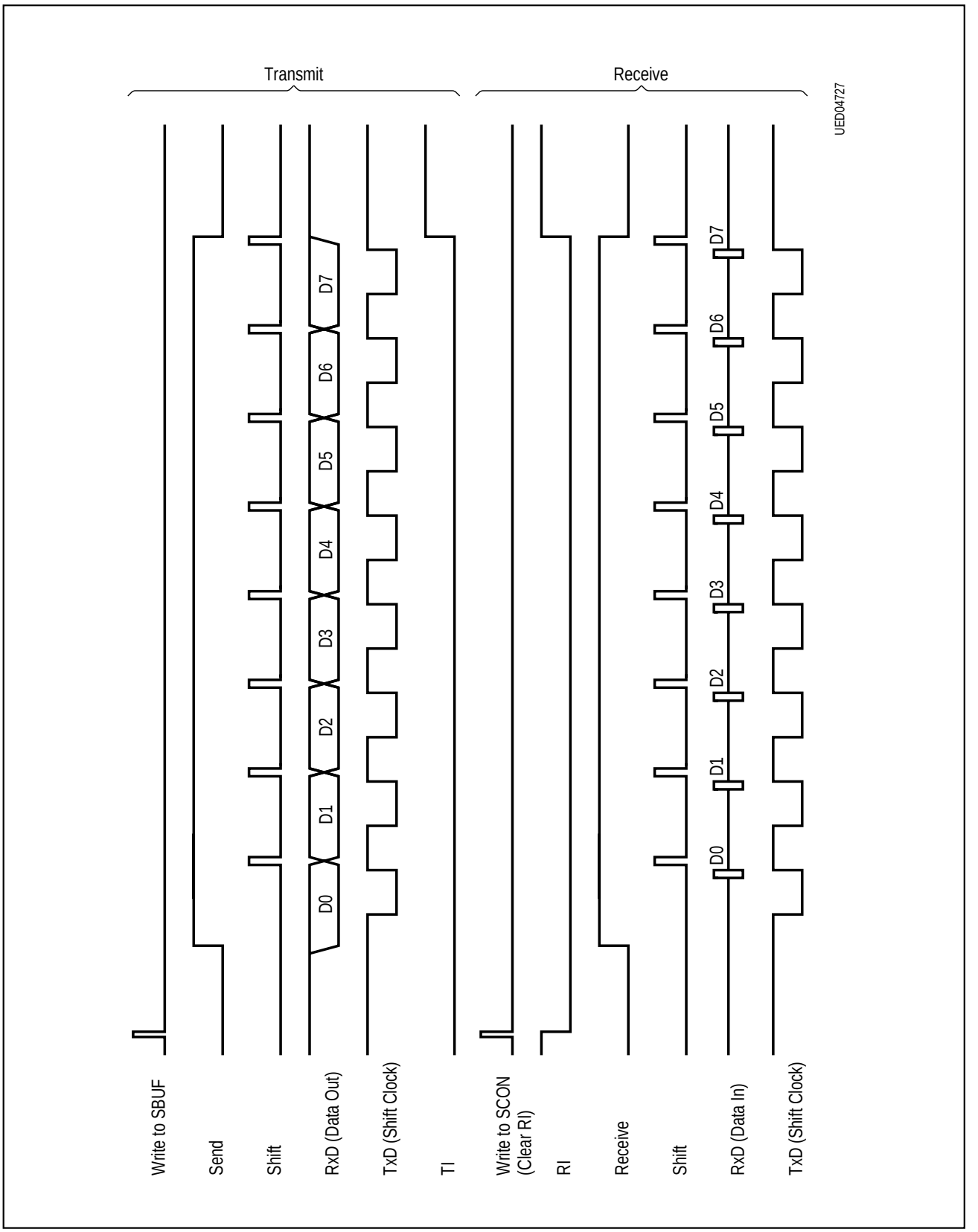


Figure 29
Serial Port Mode 0, Timing

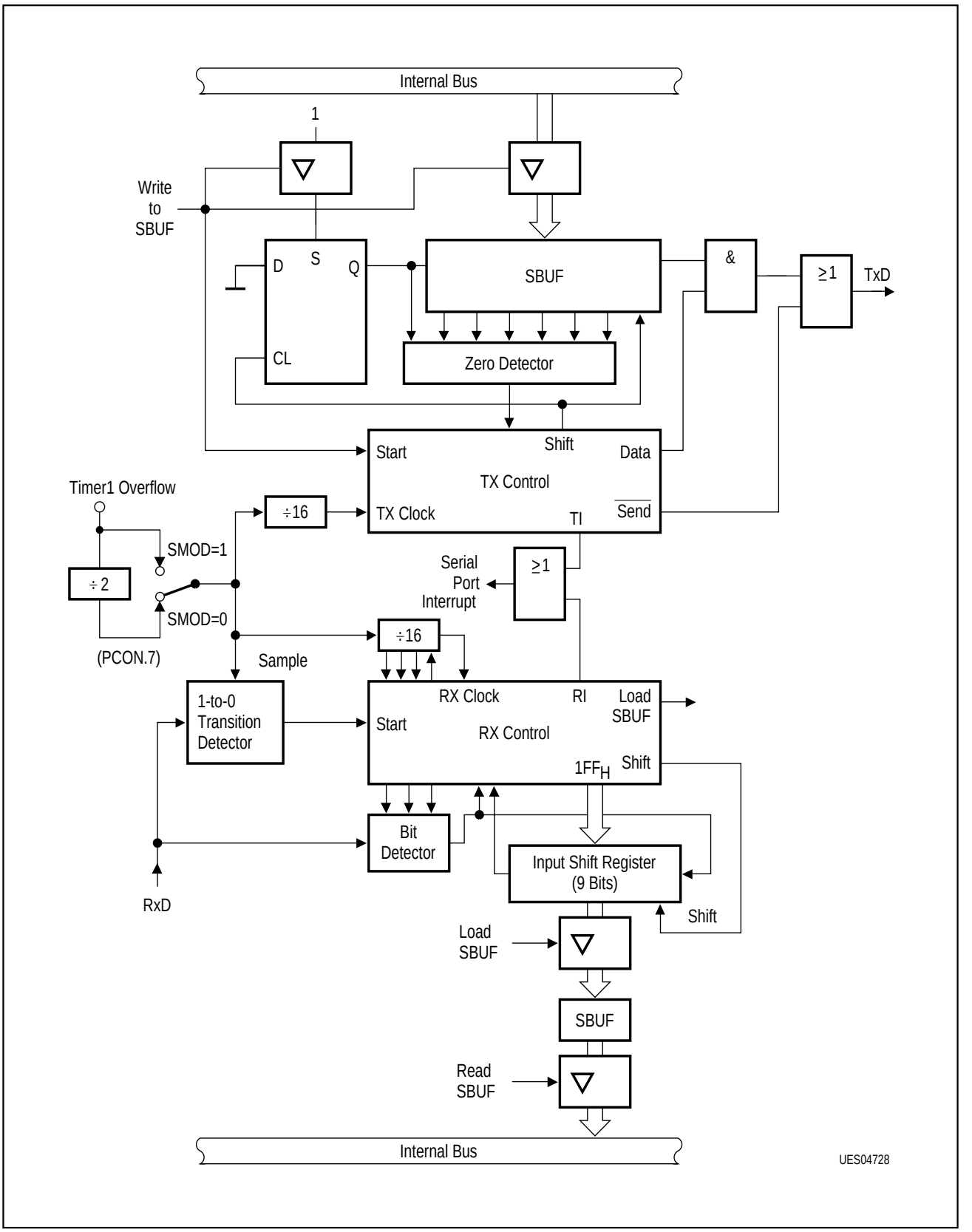


Figure 30
Serial Port Mode 1, Functional Diagram

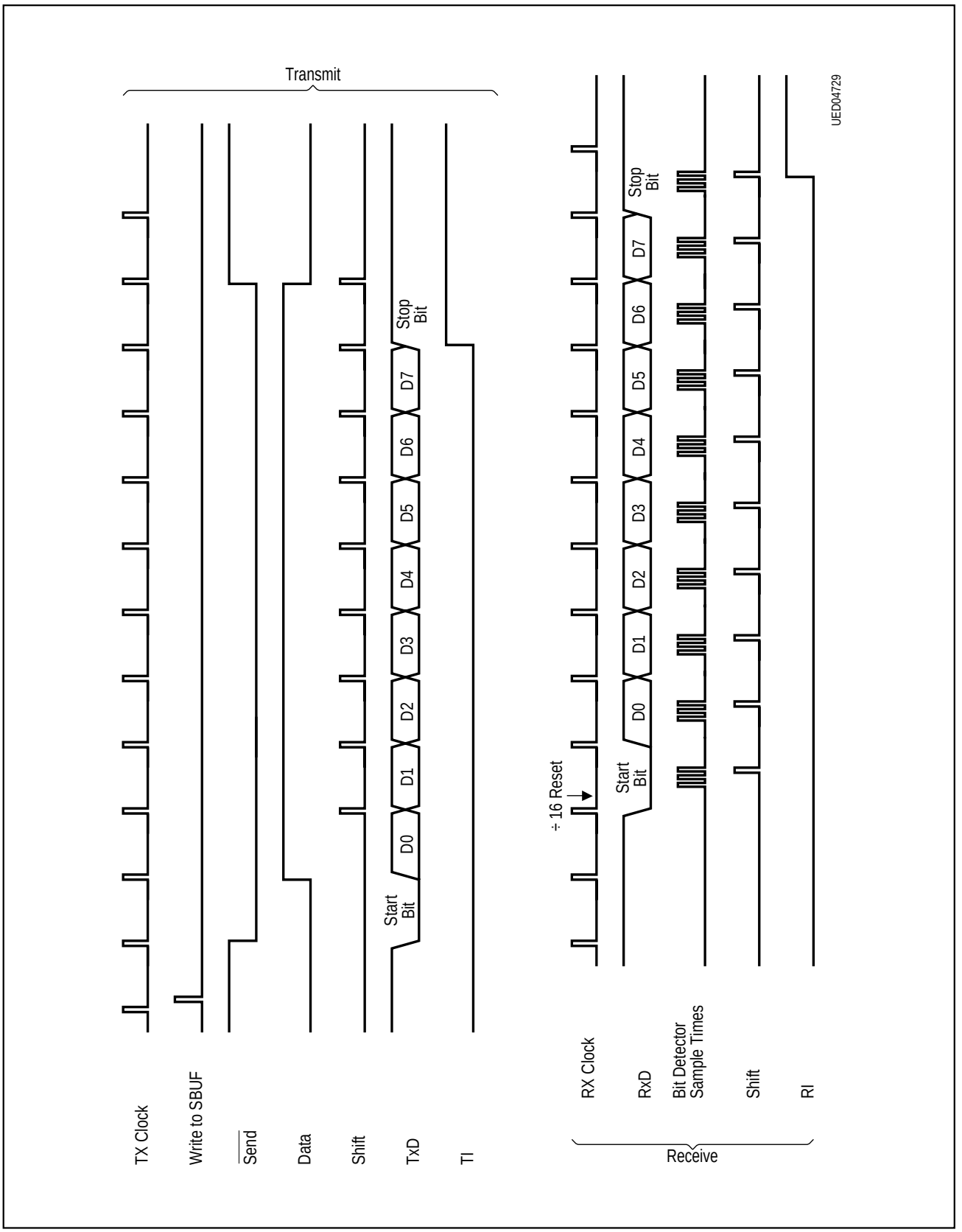


Figure 31
Serial Port Mode 1, Timing

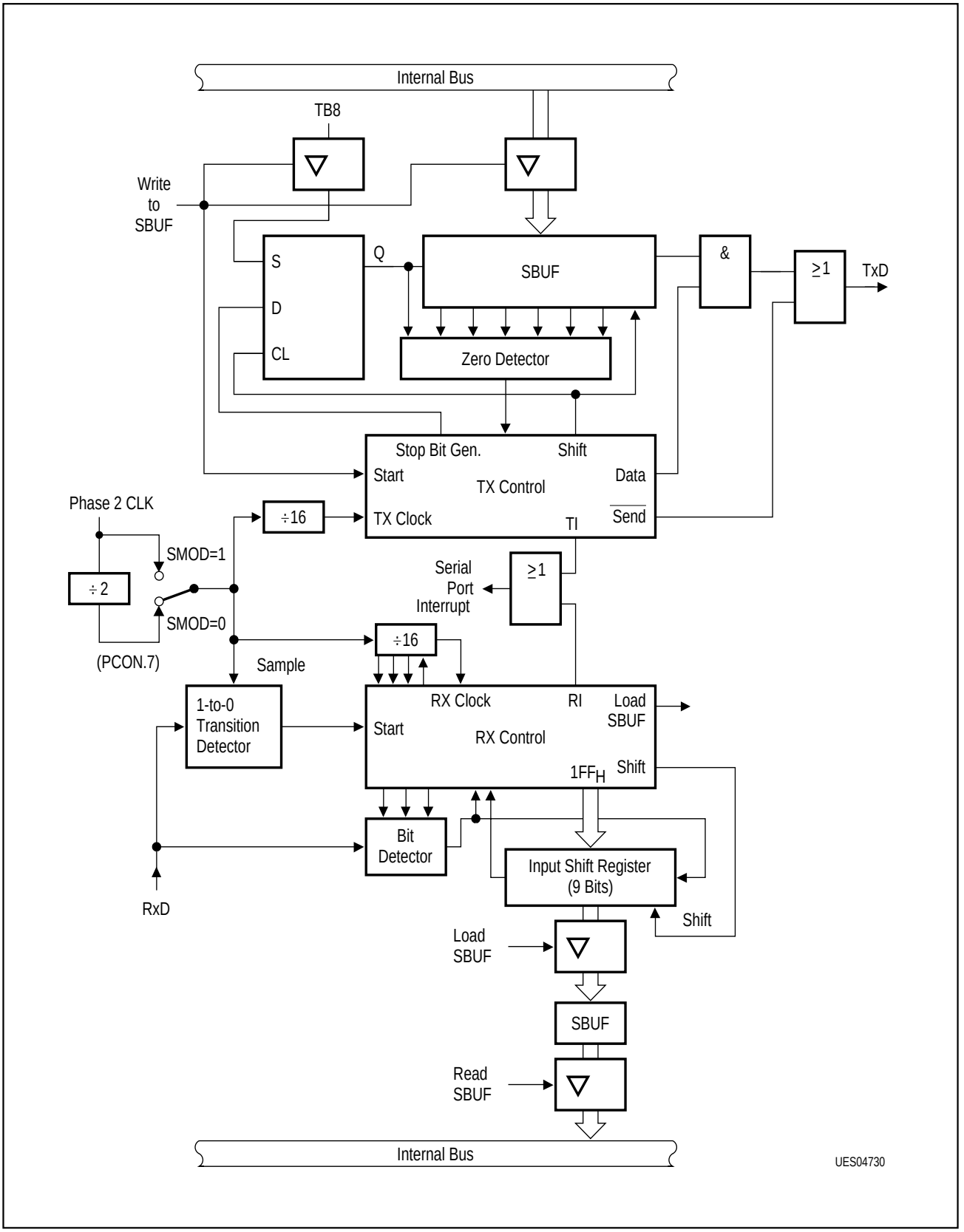


Figure 32
Serial Port Mode 2, Functional Diagram

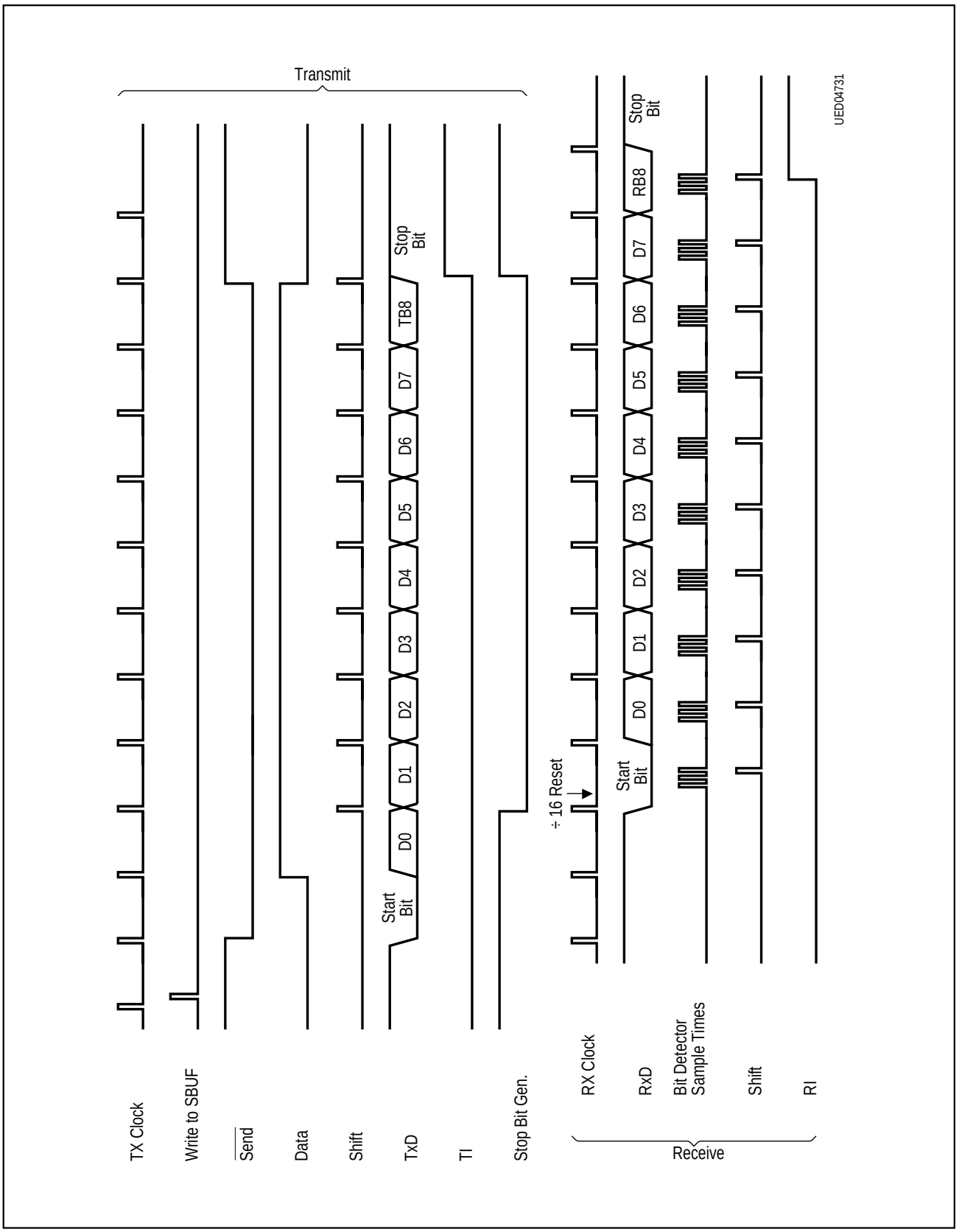


Figure 33
Serial Port Mode 2, Timing

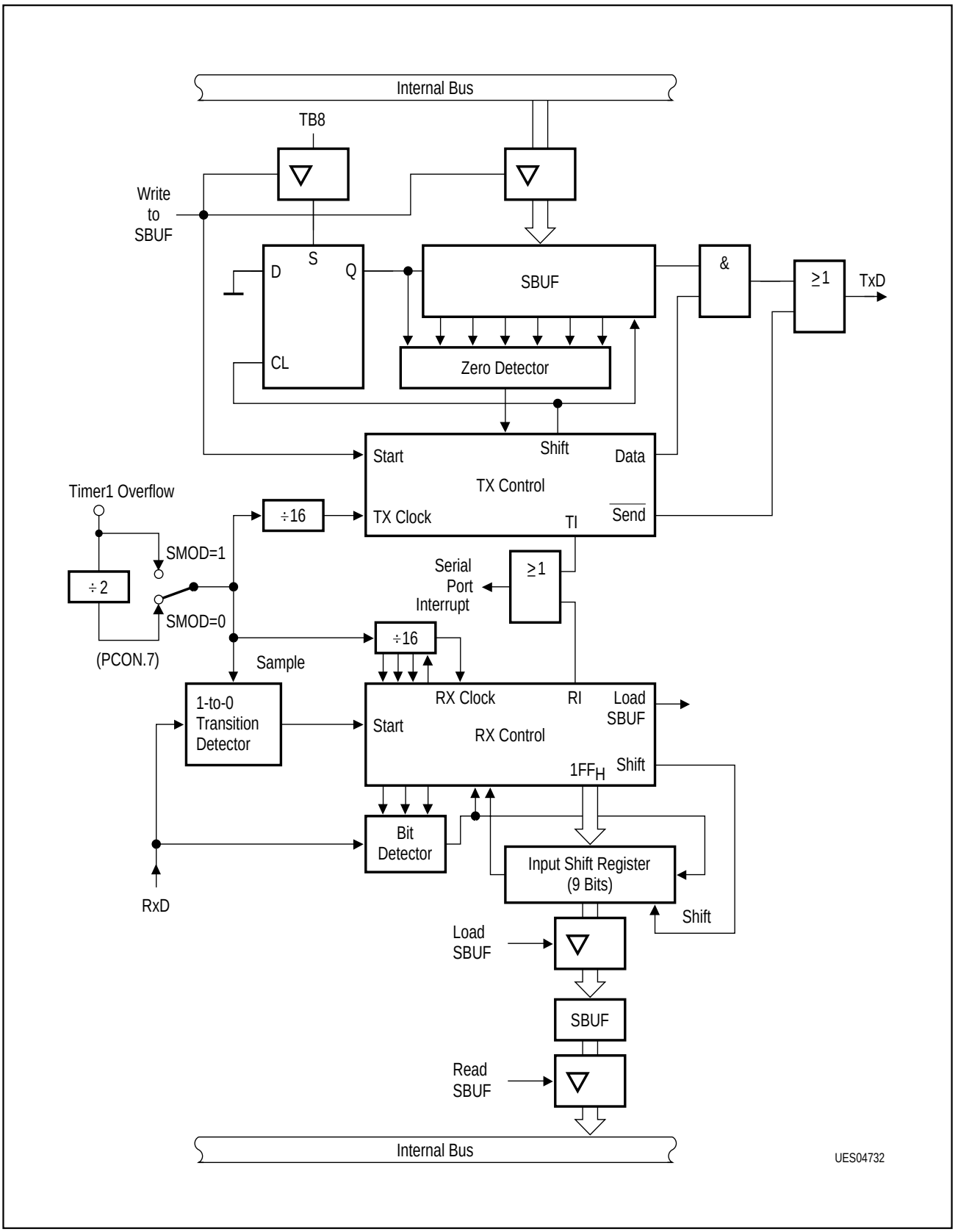


Figure 34
Serial Port Mode 3, Functional Diagram

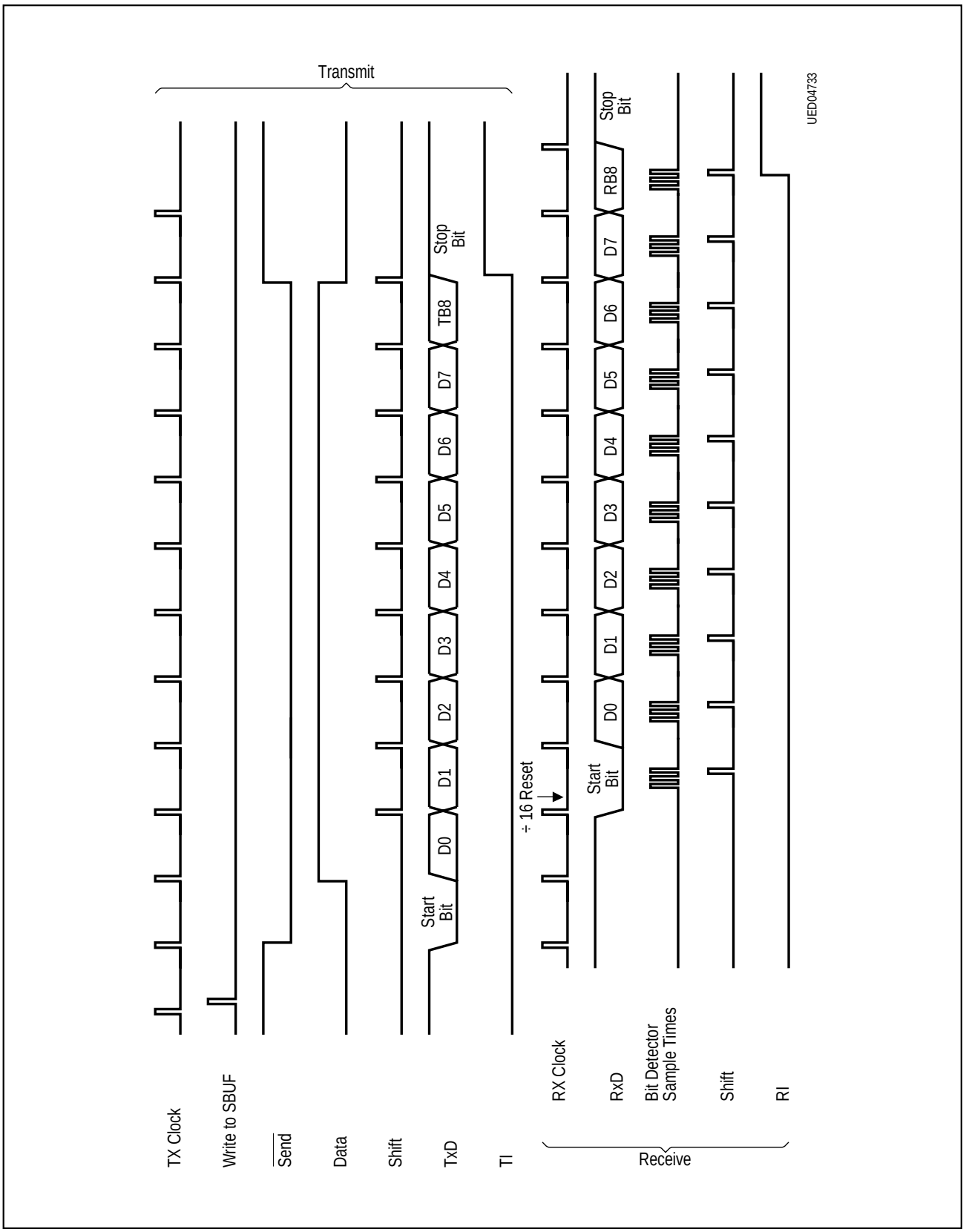


Figure 35
Serial Port Mode 3, Timing

6.3.11 Pulse Width Modulation Unit (PWM)

The on-chip-PWM unit consists of 6 quasi-8-Bit and 2 quasi-14-Bit PWM channels. Controlled via special function registers, each channel can be enabled individually.

The base frequency of an 8-Bit channel is derived from the overflow of a 6-Bit counter which counts internal clocks. On every counter overflow, the enabled PWM lines will be set to one (exception: compare values are zero) and will be reset when the 6 MSBs of the PWCOMPx-register match the counter value. To get an overall resolution of 8 bit, the high-time is stretched periodically, depending on the 2 LSBs of the PWCOMPx-register. For example, if PWCOMPx[1:0] is ‘10’, the high-time will be stretched in every second base cycle.

This type of PWM channel is called “6 plus 2”.

Table 23
Effect of PWCOMPx-Bits for 8-Bit PWM

PWCOMPx	Cycle Number ‘Stretched’
Bit 1	1,3
Bit 0	2

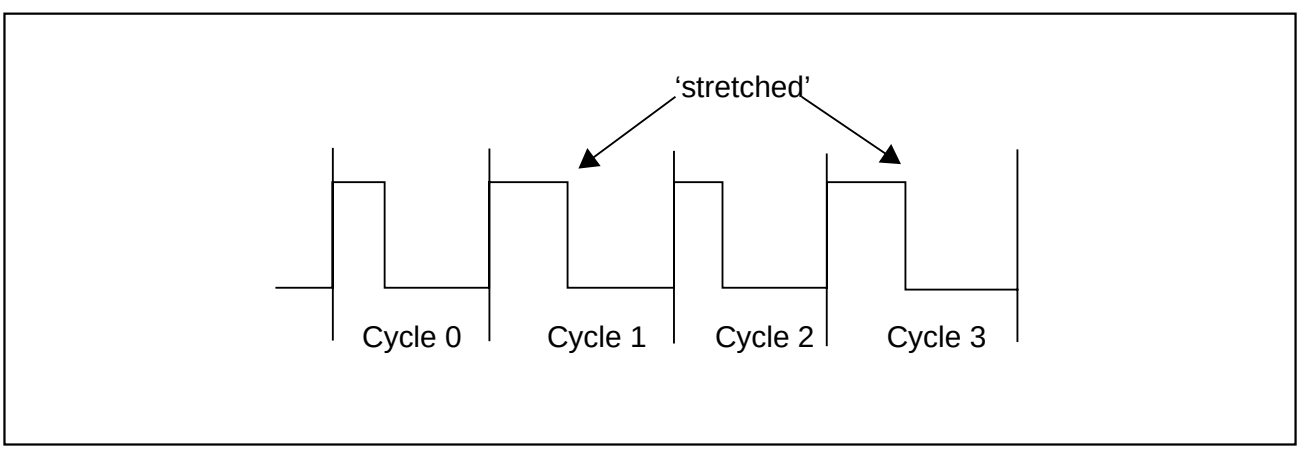


Figure 36
Simplified Example with PWCOMPx[1:0]= ‘10’

The function of an 14-Bit channel is very similar. Here, an 8-Bit counter gives the base frequency. All 8 bits of the PWCOMPx registers are compared with the counter value, and the value in PWEXTx register gives the number of stretchings within 64 successive base cycles. Thus, this type of PWM channel is called “8 plus 6”. The **Table 24** shows the influence of the PWEXTx register bits on cycles to be stretched.

Table 24
Effect of PWEXTx-Bits for 14-Bit PWM

PWEXTx	Cycle Number 'Stretched'
Bit 7	1,3,5,7,...,59,61,63
Bit 6	2,6,10,...,54,58,62
Bit 5	4,12,20,...,52,60
Bit 4	8,24,40,56
Bit 3	16,48
Bit 2	32
Bit 1	no effect
Bit 0	no effect

Table 25
Base frequencies

PWM Resolution	Base Frequency
8 bit	f_{OSC}/ 2^{CDC} x 64
14 bit	f_{OSC}/ 2^{CDC} x 256

Further Details of the PWM Unit

The PWM-output channels are placed as alternate functions to the eight lines of port 1. P1.0 ... P1.5 contain the 6 output channels with 8-bit resolution and P1.6 ... P1.7 the 2 output channels with 14-bit resolution. Each PWM-channel can be individually switched between PWM-function and port function.

The six 8-bit compare registers PWCOMP0 – PWCOMP5 are located at SFR-addresses $0F1_H$ – $0F6_H$. The two 14-bit compare registers consist each of an 8-bit register PWCOMP6 or PWCOMP7 and of a six-bit extension register PWEXT6 or PWEXT7, all located at SFR-addresses $0FA_H$ – $0FD_H$. They contain the modulation ratios of the output signals, which are related to the maximum, defined by the counter's resolution. These compare registers are double buffered and a new compare value will only be taken into the main register, after the next timer overflow or if the PWM-timer is stopped.

The PWM-timer registers located at SFR-address F7 and $F9_H$ contain the actual value of the PWM-counter low byte and high byte and can only be read by the CPU. Every compare register, which is not employed for the PWM-output can be used as an additional register. This is not allowed for register PWME.

The internal timer of the PWM unit is running as long as at least one PWM-channel is enabled by the PWM-Enable Register PWME.

PWM-Enable Register PWME

PWM-Enable Register **PWME** **SFR-Address $F8_H$**

Default after reset: 00_H

(MSB) (LSB)

E7	E6	E5	E4	E3	E2	E1	E0

E7 - E0 **= 0** The corresponding PWM-channel is disabled. P1.i functions as normal bidirectional I/O-port.

= 1 He corresponding PWM-channel is enabled. E0...E5 are channels with 8-bit resolution, while E6 and E7 are channels with 14-bit resolution.

PWM Compare Registers PWCOMP 0 - 5

PWM Compare Registers **PWCOMP 0 - 5** **SFR-Address $F1_H$-$F6_H$**

Default after reset: 00_H

(MSB) (LSB)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 7 - Bit 2 This bits define the high time of the output. If all bits are 0, the high time is 0 internal clocks. If all bits are 1, the high time is 63 internal clocks.

Bit 1 If this bit is set, every second PWM-Cycle is stretched by one internal clock, regardless of the settings of Bit7...Bit2.

Bit 0 If this bit is set, every fourth PWM-Cycle is stretched by one internal clock, regardless of the settings of Bit7...Bit2.

PWM Compare Registers PWCOMP 6, 7

PWM Compare Registers **PWCOMP 6, 7** **SFR-Address FB_H,FD_H**

Default after reset: 00_H

(MSB) (LSB)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 7 - Bit 0 This bits define the high time of the output. If all bits are 0, the high time is 0 internal clocks. If all bits are 1, the high time is 255 internal clocks.

PWM Extension Registers PWEXT 6, 7

PWM Extension Registers **PWEXT 6, 7** **SFR-Address FA_H,FC_H**

Default after reset: 00_H

(MSB) (LSB)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 7 If this bit is set, every second PWM-Cycle is stretched by one internal clock.

Bit 6 If this bit is set, every fourth PWM-Cycle is stretched by one internal clock.

Bit 5 If this bit is set, every eighth PWM-Cycle is stretched by one internal clock.

Bit 4 If this bit is set, every 16th PWM-Cycle is stretched by one internal clock.

Bit 3 If this bit is set, every 32th PWM-Cycle is stretched by one internal clock.

Bit 2 If this bit is set, every 64th PWM-Cycle is stretched by one internal clock.

Bit 1, Bit 0 This bits have to be set to 0.

Note: The described operation is independent of the setting of PWCOMP6 or PWCOMP7.
The stretch operation is interleaved between PWM-Cycles.

PWM Low Counter Registers PWCL

PWM Low Counter Registers **PWCL** **SFR-Address $F7_H$**

Default after reset: 00_H

(MSB) (LSB)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 7 - Bit 0 This bits are the low order 8 Bits of the 14 Bit PWM-Counter. This register can only be read.

PWM High Counter Registers PWCH

PWM High Counter Registers **PWCH** **SFR-Address $F9_H$**

Default after reset: 00_H

(MSB) (LSB)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Bit 7, Bit 6 This bits are undefined.

Bit 5 - Bit 0 This bits are the high order 6 Bits of the 14 Bit PWM-Counter. This register can only be read.

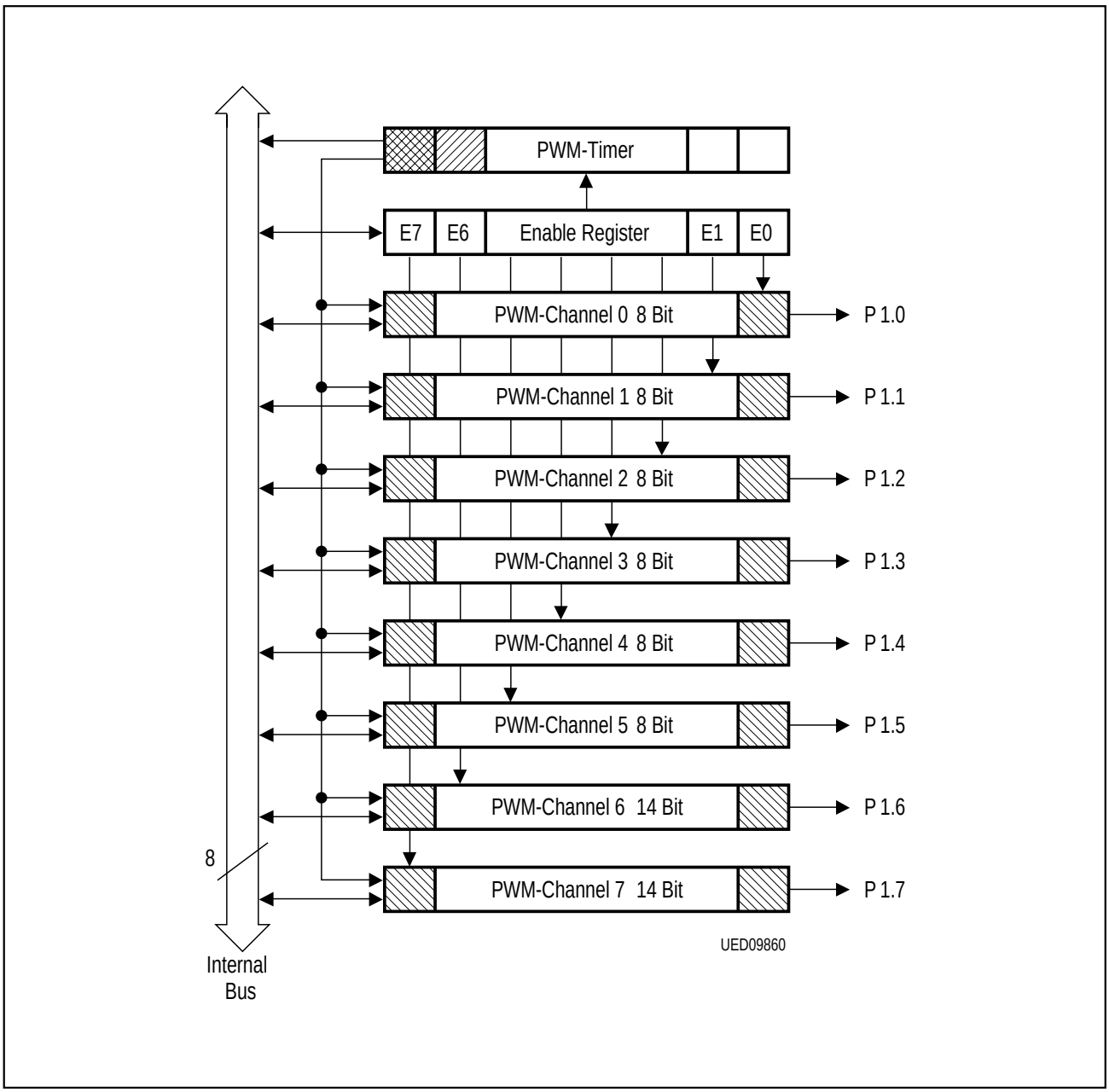


Figure 37
Block Diagram of Pulse Width Modulation Unit

6.3.12 Analog Digital Converter

The controller provides an A/D-converter with the following features:

– 4 multiplexed input channels, which can also be used as digital inputs
– 8-bit resolution
– 8.89 to 28.4 µs conversion time for 18 MHz oscillator frequency
– Analog reference voltages supplied by pins V_{DDA} and V_{SSA}

The conversion time depends on the internal master clock, used by the ADC. The clock-frequency of the internal ADC master clock is defined by the external quartz (oscillator frequency f_{OSC}), the setting of bit CDC in the Advanced Function Register AFR of the special function registers (SFR) (see **Chapter “Advanced Function Register” on page 115**), and the setting of bit PSC in the ADC Control Register ADCON (SFR). Both bits are software switches to activate or deactivate clock dividers by 2. The conversion time further depends on the sample time, adjustable by bit STADC (ADC-Control Register ADCON).

The conversion time can be calculated by:

$$t_{conversion} = \frac{(2^{2 \times STADC} + 4) \times 32 \times 2^{CDC} \times 2^{PSC}}{f_{OSC}}$$

For the conversion, the method of successive approximation via capacitor array is used. There are three user-accessible special function registers: ADCON, ADDAT and DAPR.

Special function register ADCON is used to set the operation modes, to check the status and to select one of four input channels. ADCON contains two mode bits. Bit ADM is used to choose the single or continuous conversion method. In single conversion mode (ADM = 0) only one conversion is performed after starting, while in continuous conversion mode (ADM = 1) a new conversion is automatically started on completion of the previous one. The busy flag BSY (ADCON.4) is automatically set when a conversion is in progress. After completion of the conversion it is reset by hardware. This flag can be read only, a write has no effect. MX0 and MX1 are used to select one of 4 A/D-channels. With PSC a divide by two prescaler for the internal master clock of the ADC can be activated. For PSC = 0 the internal chip-clock is used as master clock for the ADC. For PSC = 1 the internal chip-clock is divided by two before being used as master clock for the ADC. With bit STADC the sample time of the ADC can be varied. Bit STADC = 0 selects the normal sample time (sample time of 2 ADC master clock cycles), while for STADC = 1 the sample time is slowed down by a factor of 4 (sample time of 8 ADC master clock cycles) e.g. for high-impedance input signals.

The special function register ADDAT holds the converted digital 8-bit data result. The data remains in ADDAT until it is overwritten by the next converted data. ADDAT can be read or written under software control. A start of conversion is triggered by a write-to DAPR instruction. The data written must be 00_H.

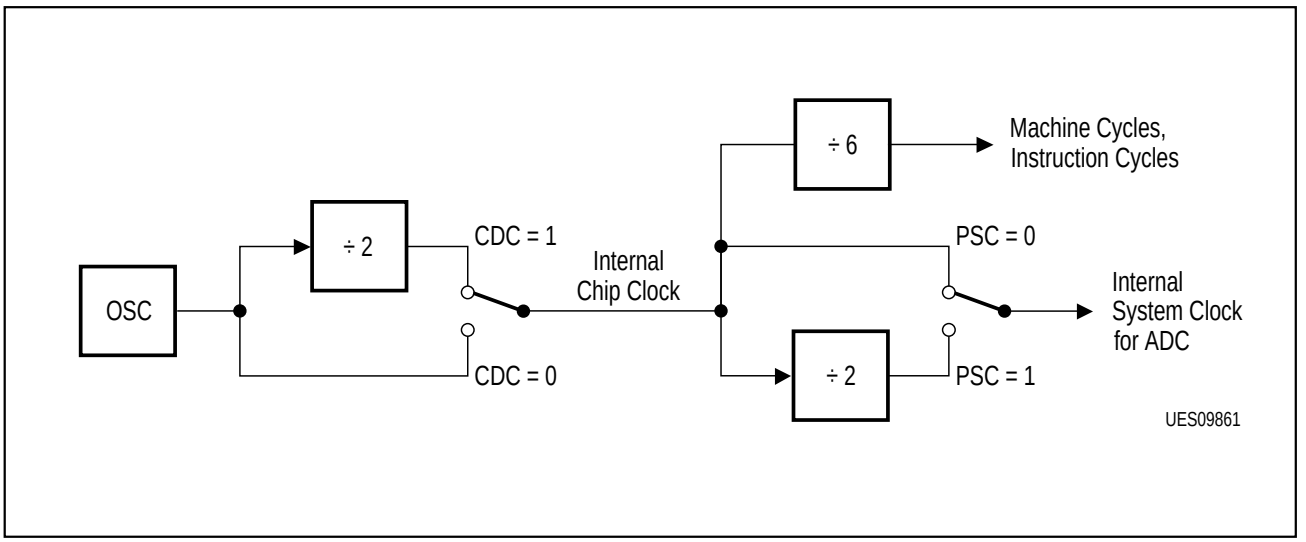


Figure 38
Internal System Clock of the ADC

ADC-Start Register DAPR

ADC-Start Register **DAPR** **SFR-Address DA_H**

Default after reset: 00_H

(MSB)							(LSB)
–	–	–	–	–	–	–	–

Only the address of DAPR is used to decode a start-of-conversion signal. No bits are implemented. A read from DAPR might show random values.

ADC-Control Register ADCON

ADC-Control Register **ADCON** **SFR-Address $D8_H$**

Default after reset: 00_H

(MSB) (LSB)

PSC	STADC	IADC	BSY	ADM	0	MX1	MX0

This register is bit addressable.

PSC Prescaler control: PSC = 0 for prescaler not active. Internal master clock of ADC is equal to the internal chip clock. PSC = 1 for prescaler active. Internal master clock of ADC is at half the internal chip clock.

STADC ADC sample time adjustment: STADC = 0 for normal sample time. STADC = 1 for fourfold sample time.

IADC ADC interrupt request flag. Set on completion of AD-conversion. Must be cleared by software.

BSY Busy flag; = 1, during conversion.

ADM ADC-conversion mode: ADM = 0 for single and ADM = 1 for continuous conversion.

ADCON.2 Always to be written with ‘0’.

MX1, MX0 ADC-channel select.

Table 26
ADC-Channel Select

MX1	MX0	Selected Channel
0	0	0
0	1	1
1	0	2
1	1	3

ADC-Data Register ADDAT

ADC-Data Register **ADDAT** **SFR-Address $D9_H$**

Default after reset: undefined

(MSB)							(LSB)
AD7	**AD6**	**AD5**	**AD4**	**AD3**	**AD2**	**AD1**	**AD0**

AD (7-0) 8-Bit Analog Data Value

6.3.13 Advanced Function Register

Advanced Function Register **AFR** **SFR-Address $A6_H$**

Default after reset: 00xxxxxxB

(MSB)							(LSB)
CDC	**WDT**	**0**	**0**	**0**	**0**	**0**	**0**

CDC Clock divider control bit. If set, the clock divider is on. The internal clock frequency is half the external oscillator frequency. If cleared, the clock divider is off. The internal clock frequency is equal to the external oscillator frequency. This feature can be used to reduce power dissipation by reducing the internal clock frequency by a factor of two.

WDT See **Chapter “Watchdog Timer” on page 87**.

AFR.0 – AFR.5 Reserved, always to be written with ‘0’.

The machine cycle time is controlled by bit CDC too. For CDC = 1 a machine cycle consists of 12 oscillator cycles and for CDC = 0 of six oscillator cycles (see **Figure 14**).

6.3.14 Instruction Set

The assembly language uses the same instruction set and the same instruction opcodes as the 8051 microcomputer family.

6.3.14.1 Notes on Data Addressing Modes

Rn	–	Working register R0 – R7.
direct	–	128 internal RAM-locations, any I/O-port, control or status register.
@Ri	–	Indirect internal RAM-location addressed by register R0 or R1.
#data	–	8-bit constant included in instruction.
#data 16	–	16-bit constant included as bytes 2 & 3 of instruction.
bit	–	128 software flags, any I/O-pin, control or status bit in special function registers.

Operations working on external data memory (MOVX …) are used to access the extended internal data RAM (XRAM).

6.3.14.2 Notes on Program Addressing Modes

addr 16	–	Destination address for LCALL & LJMP may be anywhere within the program memory address space.
addr 11	–	Destination address for ACALL & AJMP will be within the same 2 Kbyte of the following instruction.
rel	–	SJMP and all conditional jumps include an 8-bit offset byte. Range is + 127/ – 128 bytes relative to first byte of the following instruction.

6.3.14.3 Instruction Set Description

Table 27
Arithmetic Operations

Mnemonic		Description	Byte
ADD	A, Rn	Add register to Accumulator	1
ADD	A, direct	Add direct byte to Accumulator	2
ADD	A, @Ri	Add indirect RAM to Accumulator	1
ADD	A, #data	Add immediate data to Accumulator	2
ADDC	A, Rn	Add register to Accumulator with Carry flag	1
ADDC	A, direct	Add direct byte to A with Carry flag	2
ADDC	A, @Ri	Add indirect RAM to A with Carry flag	1
ADDC	A, #data	Add immediate data to A with Carry flag	2
SUBB	A, Rn	Subtract register from A with Borrow	1
SUBB	A, direct	Subtract direct byte from A with Borrow	2
SUBB	A, @Ri	Subtract indirect RAM from A with Borrow	1
SUBB	A, #data	Subtract immediate data from A with Borrow	2
INC	A	Increment Accumulator	1
INC	Rn	Increment register	1
INC	direct	Increment direct byte	2
INC	@Ri	Increment indirect RAM	1
DEC	A	Decrement Accumulator	1
DEC	Rn	Decrement register	1
DEC	direct	Decrement direct byte	2
DEC	@Ri	Decrement indirect RAM	1
INC	DPTR	Increment Data Pointer	1
MUL	AB	Multiply A & B	1
DIV	AB	Divide A & B	1
DA	A	Decimal Adjust Accumulator	1

Table 28
Logical Operations

Mnemonic		Description	Byte
ANL	A, Rn	AND register to Accumulator	1
ANL	A, direct	AND direct byte to Accumulator	2
ANL	A, @Ri	AND indirect RAM to Accumulator	1
ANL	A, #data	AND immediate data to Accumulator	2
ANL	direct, A	AND Accumulator to direct byte	2
ANL	direct, #data	AND immediate data to direct byte	3
ORL	A, Rn	OR register to Accumulator	1
ORL	A, direct	OR direct byte to Accumulator	2
ORL	A, @Ri	OR indirect RAM to Accumulator	1
ORL	A, #data	OR immediate data to Accumulator	2
ORL	direct, A	OR Accumulator to direct byte	2
ORL	direct, #data	OR immediate data to direct byte	3
XRL	A, Rn	Exclusive-OR register to Accumulator	1
XRL	A, direct	Exclusive-OR direct byte to Accumulator	2
XRL	A, @Ri	Exclusive-OR indirect RAM to Accumulator	1
XRL	A, #data	Exclusive-OR immediate data to Accumulator	2
XRL	direct, A	Exclusive-OR Accumulator to direct byte	2
XRL	direct, #data	Exclusive-OR immediate data to direct	3
CLR	A	Clear Accumulator	1
CPL	A	Complement Accumulator	1
RL	A	Rotate Accumulator left	1
RLC	A	Rotate A left through the Carry flag	1
RR	A	Rotate Accumulator right	1
RRC	A	Rotate A right through Carry flag	1
SWAP	A	Swap nibbles within the Accumulator	1

Table 29
Data Transfer Operations

Mnemonic		Description	Byte
MOV	A, Rn	Move register to Accumulator	1
MOV	A, direct	Move direct byte to Accumulator	2
MOV	A, @Ri	Move indirect RAM to Accumulator	1
MOV	A, #data	Move immediate data to Accumulator	2
MOV	Rn, A	Move Accumulator to register	1
MOV	Rn, direct	Move direct byte to register	2
MOV	Rn, #data	Move immediate data to register	2
MOV	direct, A	Move Accumulator to direct byte	2
MOV	direct, Rn	Move register to direct byte	2
MOV	direct, direct	Move direct byte to direct	3
MOV	direct, @Ri	Move indirect RAM to direct byte	2
MOV	direct, #data	Move immediate data to direct byte	3
MOV	@Ri, A	Move Accumulator to indirect RAM	1
MOV	@Ri, direct	Move direct byte to indirect RAM	2
MOV	@Ri, #data	Move immediate data to indirect RAM	2
MOV	DPTR, #data 16	Load Data Pointer with a 16-bit constant	3
MOVC	A@A + DPTR	Move Code byte relative to DPTR to Accumulator	1
MOVC	A@A + PC	Move Code byte relative to PC to Accumulator	1
MOVX	A, @Ri	Move External RAM (8-bit addr) to Accumulator[1)]	1
MOVX	A, @DPTR	Move External RAM (16-bit addr) to Accumulator	1
MOVX	@Ri, A	Move A to External RAM (8-bit addr)[1)]	1
MOVX	@DPTR, A	Move A to External RAM (16-bit addr)	1
PUSH	direct	Push direct byte onto stack	2
POP	direct	Pop direct byte from stack	2
XCH	A, Rn	Exchange register with Accumulator	1
XCH	A, direct	Exchange direct byte with Accumulator	2
XCH	A, @Ri	Exchange indirect RAM with Accumulator	1
XCHD	A, @Ri	Exchange low-order digital indirect RAM with A[1)]	1

[1)] not applicable for the SDA525x

Table 30
Boolean Variable Manipulation

Mnemonic		Description	Byte
CLR	C	Clear Carry flag	1
CLR	bit	Clear direct bit	2
SETB	C	Set Carry flag	1
SETB	bit	Set direct bit	2
CPL	C	Complement Carry flag	1
CPL	bit	Complement direct bit	2
ANL	C, bit	AND direct bit to Carry flag	2
ANL	C, /bit	AND complement of direct bit to Carry	2
ORL	C, bit	OR direct bit to Carry flag	2
ORL	C, /bit	OR complement of direct bit to Carry	2
MOV	C, bit	Move direct bit to Carry flag	2
MOV	bit, C	Move Carry flag to direct bit	2

Table 31
Program and Machine Control Operations

Mnemonic	Description	Byte
ACALL addr 11	Absolute subroutine call	2
LCALL addr 16	Long subroutine call	3
RET	Return from subroutine	1
RETI	Return from interrupt	1
AJMP addr 11	Absolute jump	2
LJMP addr 16	Long jump	3
SJMP rel	Short jump (relative addr)	2
JMP @A + DPTR	Jump indirect relative to the DPTR	1
JZ rel	Jump if Accumulator is zero	2
JNZ rel	Jump if Accumulator is not zero	2
JC rel	Jump if Carry flag is set	2
JNC rel	Jump if Carry flag is not set	2
JB bit, rel	Jump if direct bit set	3
JNB bit, rel	Jump if direct bit not set	3
JBC bit, rel	Jump if direct bit is set and clear bit	3
CJNE A, direct rel	Compare direct to A and jump if not equal	3
CJNE A, #data, rel	Compare immediate to A and jump if not equal	3
CJNE Rn, #data, rel	Compare immediate to register and jump if not equal	3
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal	3
DJNZ Rn, rel	Decrement register and jump if not zero	2
DJNZ direct, rel	Decrement direct and jump if not zero	3
NOP	No operation	1

6.3.15 Instruction Opcodes in Hexadecimal Order

Table 32
Instruction Opcodes in Hexadecimal Order

Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	code addr
02	3	LJMP	code addr
03	1	RR	A
04	1	INC	A
05	2	INC	data addr
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	bit addr, code addr
11	2	ACALL	code addr
12	3	LCALL	code addr
13	1	RRC	A
14	1	DEC	A
15	2	DEC	data addr
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	bit addr, code addr
21	2	AJMP	code addr
22	1	RET	
23	1	RL	A
24	2	ADD	A, #data

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
25	2	ADD	A, data addr
26	1	ADD	A, @R0
27	1	ADD	A, @R1
28	1	ADD	A, R0
29	1	ADD	A, R1
2A	1	ADD	A, R2
2B	1	ADD	A, R3
2C	1	ADD	A, R4
2D	1	ADD	A, R5
2E	1	ADD	A, R6
2F	1	ADD	A, R7
30	3	JNB	bit addr, code addr
31	2	ACALL	code addr
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A, #data
35	2	ADDC	A, data addr
36	1	ADDC	A, @R0
37	1	ADDC	A, @R1
38	1	ADDC	A, R0
39	1	ADDC	A, R1
3A	1	ADDC	A, R2
3B	1	ADDC	A, R3
3C	1	ADDC	A, R4
3D	1	ADDC	A, R5
3E	1	ADDC	A, R6
3F	1	ADDC	A, R7
40	2	JC	code addr
41	2	AJMP	code addr
42	2	ORL	data addr., A
43	3	ORL	data addr, #data
44	2	ORL	A, #data
45	2	ORL	A, data addr
46	1	ORL	A, @R0
47	1	ORL	A, @R1
48	1	ORL	A, R0
49	1	ORL	A, R1
4A	1	ORL	A, R2
4B	1	ORL	A, R3

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
4C	1	ORL	A, R4
4D	1	ORL	A, R5
4E	1	ORL	A, R6
4F	1	ORL	A, R7
50	2	JNC	code addr
51	2	ACALL	code addr
52	2	ANL	data addr, A
53	3	ANL	data addr, #data
54	2	ANL	A, #data
55	2	ANL	A, data addr
56	1	ANL	A, @R0
57	1	ANL	A, @R1
58	1	ANL	A, R0
59	1	ANL	A, R1
5A	1	ANL	A, R2
5B	1	ANL	A, R3
5C	1	ANL	A, R4
5D	1	ANL	A, R5
5E	1	ANL	A, R6
5F	1	ANL	A, R7
60	2	JZ	code addr
61	2	AJMP	code addr.
62	2	XRL	data addr, A
63	3	XRL	data addr, #data
64	2	XRL	A, #data
65	2	XRL	A, data addr
66	1	XRL	A, @R0
67	1	XRL	A, @R1
68	1	XRL	A, R0
69	1	XRL	A, R1
6A	1	XRL	A, R2
6B	1	XRL	A, R3
6C	1	XRL	A, R4
6D	1	XRL	A, R5
6E	1	XRL	A, R6
6F	1	XRL	A, R7
70	2	JNZ	code addr
71	2	ACALL	code addr
72	2	ORL	C, bit addr

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
73	1	JMP	@A + DPTR
74	2	MOV	A, #data
75	3	MOV	data addr, #data
76	2	MOV	@R0, #data
77	2	MOV	@R1, #data
78	2	MOV	R0, #data
79	2	MOV	R1, #data
7A	2	MOV	R2, #data
7B	2	MOV	R3, #data
7C	2	MOV	R4, #data
7D	2	MOV	R5, #data
7E	2	MOV	R6, #data
7F	2	MOV	R7, #data
80	2	SJMP	code addr
81	2	AJMP	code addr
82	2	ANL	C, bit addr
83	1	MOVC	A, @A + PC
84	1	DIV	AB
85	3	MOV	data addr, data addr
86	2	MOV	data addr, @R0
87	2	MOV	data addr, @R1
88	2	MOV	data addr, R0
89	2	MOV	data addr, R1
8A	2	MOV	data addr, R2
8B	2	MOV	data addr, R3
8C	2	MOV	data addr, R4
8D	2	MOV	data addr, R5
8E	2	MOV	data addr, R6
8F	2	MOV	data addr, R7
90	3	MOV	DPTR, #data 16
91	2	ACALL	code addr
92	2	MOV	bit addr, C
93	1	MOVC	A, @A + DPTR
94	2	SUBB	A, #data
95	2	SUBB	A, data addr
96	1	SUBB	A, @R0
97	1	SUBB	A, @R1
98	1	SUBB	A, R0
99	1	SUBB	A, R1

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
9A	1	SUBB	A, R2
9B	1	SUBB	A, R3
9C	1	SUBB	A, R4
9D	1	SUBB	A, R5
9E	1	SUBB	A, R6
9F	1	SUBB	A, R7
A0	2	ORL	C, /bit addr
A1	2	AJMP	code addr
A2	2	MOV	C, bit addr
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0, data addr
A7	2	MOV	@R1, data addr
A8	2	MOV	R0, data addr
A9	2	MOV	R1, data addr
AA	2	MOV	R2, data addr
AB	2	MOV	R3, data addr
AC	2	MOV	R4, data addr
AD	2	MOV	R5, data addr
AE	2	MOV	R6, data addr
AF	2	MOV	R7, data addr
B0	2	ANL	C, /bit addr
B1	2	ACALL	code addr
B2	2	CPL	bit addr
B3	1	CPL	C
B4	3	CJNE	A, #data, code addr
B5	3	CJNE	A, data addr, code addr
B6	3	CJNE	@R0, #data, code addr
B7	3	CJNE	@R1, #data, code addr
B8	3	CJNE	R0, #data, code addr
B9	3	CJNE	R1, #data, code addr
BA	3	CJNE	R2, #data, code addr
BB	3	CJNE	R3, #data, code addr
BC	3	CJNE	R4, #data, code addr
BD	3	CJNE	R5, #data, code addr
BE	3	CJNE	R6, #data, code addr
BF	3	CJNE	R7, #data, code addr
C0	2	PUSH	data addr

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
C1	2	AJMP	code addr
C2	2	CLR	bit addr
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A, data addr
C6	1	XCH	A, @R0
C7	1	XCH	A, @R1
C8	1	XCH	A, R0
C9	1	XCH	A, R1
CA	1	XCH	A, R2
CB	1	XCH	A, R3
CC	1	XCH	A, R4
CD	1	XCH	A, R5
CE	1	XCH	A, R6
CF	1	XCH	A, R7
D0	2	POP	data addr
D1	2	ACALL	code addr
D2	2	SETB	bit addr
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	data addr, code addr
D6		not applicable	
D7		not applicable	
D8	2	DJNZ	R0, code addr
D9	2	DJNZ	R1, code addr
DA	2	DJNZ	R2, code addr
DB	2	DJNZ	R3, code addr
DC	2	DJNZ	R4, code addr
DD	2	DJNZ	R5, code addr
DE	2	DJNZ	R6, code addr
DF	2	DJNZ	R7, code addr
E0	1	MOVX	A, @DPTR
E1	2	AJMP	code addr
E2		not applicable	
E3		not applicable	
E4	1	CLR	A
E5	2	MOV	A, data addr
E6	1	MOV	A, @R0
E7	1	MOV	A, @R1

Table 32
Instruction Opcodes in Hexadecimal Order (cont'd)

Hex Code	Number of Bytes	Mnemonic	Operands
E8	1	MOV	A, R0
E9	1	MOV	A, R1
EA	1	MOV	A, R2
EB	1	MOV	A, R3
EC	1	MOV	A, R4
ED	1	MOV	A, R5
EE	1	MOV	A, R6
EF	1	MOV	A, R7
F0	1	MOVX	@DPTR, A
F1	2	ACALL	code addr
F2		not applicable	
F3		not applicable	
F4	1	CPL	A
F5	2	MOV	data addr, A
F6	1	MOV	@R0, A
F7	1	MOV	@R1, A
F8	1	MOV	R0, A
F9	1	MOV	R1, A
FA	1	MOV	R2, A
FB	1	MOV	R3, A
FC	1	MOV	R4, A
FD	1	MOV	R5, A
FE	1	MOV	R6, A
FF	1	MOV	R7, A

7 Electrical Characteristics

7.1 Absolute Maximum Ratings

Table 33

Parameter	Symbol	Limit Values	Unit
Voltage on any pin with respect to ground (V_{SS})	V_S	– 0.5 to 7	V
Power dissipation	P_{tot}	1	W
Ambient temperature under bias	T_A	0 to 70	°C
Storage temperature	T_{stg}	– 65 to 125	°C

7.2 DC-Characteristics

Table 34
DC-Characteristics

T_A = 0 to 70 °C; V_{DD} = 5 V ± 10 %, V_{SS} = 0 V(C_L = 80 pF)

Parameter	Symbol	Limit Values		Units	Test Condition
		min.	max.		
L-input voltage (all except SC)	V_{IL}	– 0.5	0.8	V	
H-input voltage (all except XTAL1, SC)	V_{IH}	2.0	V_{DD} + 0.5	V	
H-input voltage (XTAL1,LCIN)	V_{IH1}	0.7 V_{DD}	V_{DD} + 0.5	V	
L-output voltage	V_{OL}	–	0.45	V	I_{OL} = 3.2 mA
H-output voltage (ports 1 – 4 in port-mode)	V_{OH}	2.4	–	V	I_{OH} = – 40 µA
H-output voltage (all except ports in port-mode)	V_{OH1}	2.4	–	V	I_{OH} = –1.6 mA
Logical 0 input current (ports 1 – 4, $\overline{RST}$,VS)	I_{IL1}		– 50	µA	V_{IN} = 0.45 V
Input leakage current (port 0, port 2, HS/SC)	I_{LI}		± 1	µA	0.45 V ≤ V_{IN} ≤ V_{DD}
Power supply current (Sum of V_{DD}- and V_{DDA}-Pins)	I_{DD}		85	mA	V_{DD} = 5 V; f_{OSC} = 18 MHz
Idle current (Sum of V_{DD}- and V_{DDA}-Pins)	I_{IDLE}		45	mA	V_{DD} = 5 V; f_{OSC} = 18 MHz

Table 34
DC-Characteristics (cont'd)

T_A = 0 to 70 °C; V_{DD} = 5 V ± 10 %, V_{SS} = 0 V(C_L = 80 pF)

Parameter	Symbol	Limit Values		Units	Test Condition
		min.	max.		
Power down current (Sum of V_{DD}- and V_{DDA}-Pins)	I_{PD}		1.5	mA	V_{DD} = 5 V
Pin capacitance	C_{IO}		10	pF	f_C = 1 MHz
H-SC voltage	V_{SCH}	1)	V_{DD} + 0.5	V	
L-SC voltage 1	V_{SCL1}	– 0.5	1)	V	
L-SC voltage 2	V_{SCL2}	1)	1)	V	
Analog input capacitance	C_I		45	pF	
ADC-total unadjusted error	TUE		t.b.d.	LSB	
Analog ground voltage	V_{SSA}	V_{SS}	V_{SS}	V	
Analog reference voltage	V_{DDA}	V_{DD}	V_{DD}	V	
Analog input voltage	V_{AI}	V_{SS} – 0.2	V_{DD} + 0.2	V	
Video input signal level	V_{CVBS}	0.7	2.0	V	
Synchron signal amplitude	V_{SYNC}	0.2	1.0	V	
Data amplitude	V_{DAT}	0.3	1.0	V	

1) adjustable, see **Chapter "Sandcastle Decoder" on page 33** and **figure 41**.

7.3 AC-Characteristics

External Clock Drive XTAL1 / Quartz Clock Drive XTAL1 - XTAL2

Table 35

T_A = 0 to 70 °C; V_{DD} = 5 V ± 10 %, V_{SS} = 0 V (C_L = 80 pF)

Parameter	Symbol	Limit Values		Unit
		Fixed internal Clock $1/t_{CLCL}$ = 18 MHz		
		min.	max.	
Cycle time	t_{CY}	6 t_{CLCL}	–	ns
Address out to valid instr in	t_{AVIV}	–	2.3. x t_{CLCL}	ns
Oscillator period	t_{CLCL}	55.6	–	ns
External clock high time	t_{CHCX}	13	–	ns
External clock low time	t_{CLCX}	13	–	ns
External clock rise time	t_{CLCH}	–	13	ns
External clock fall time	t_{CHCL}	–	13	ns

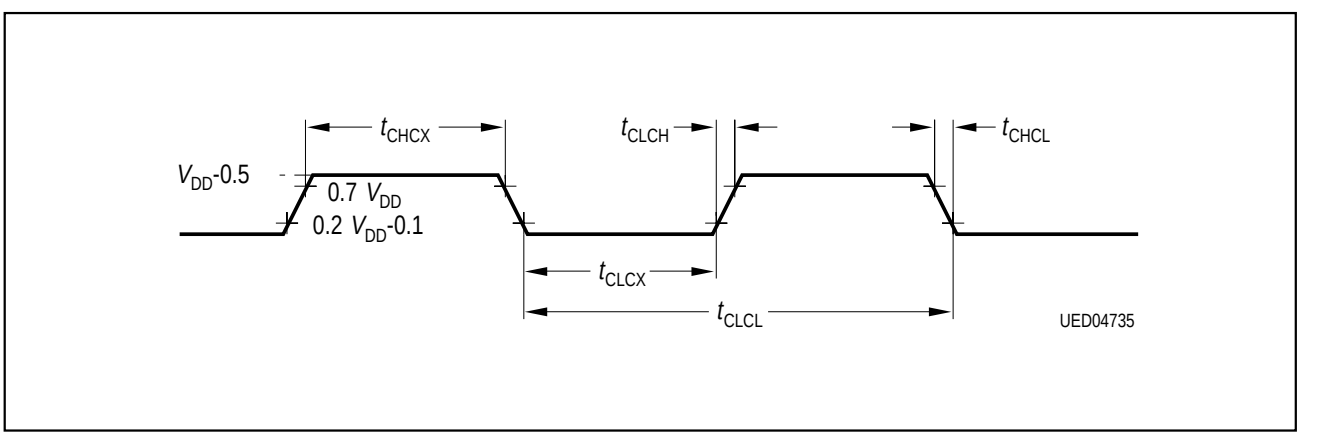


Figure 39
External Clock Cycle

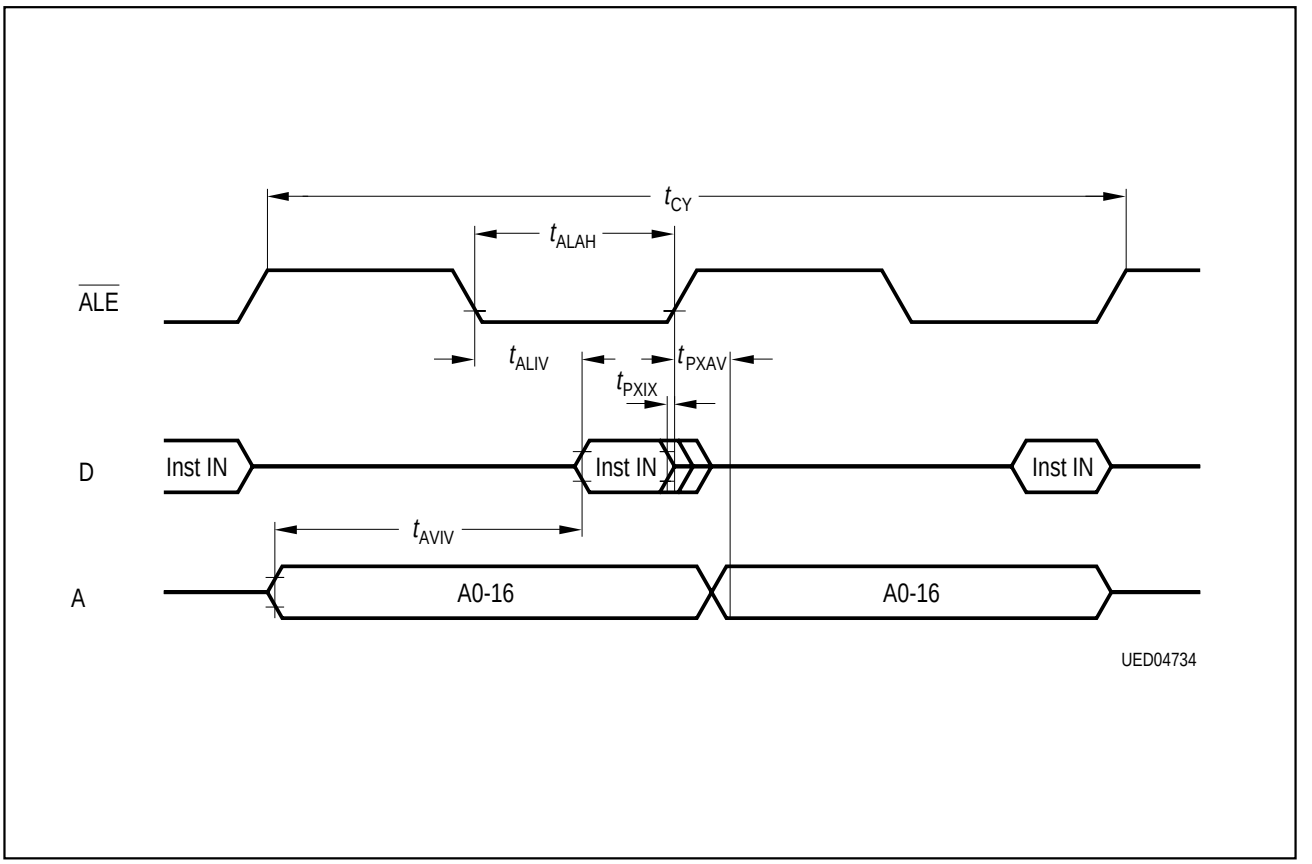


Figure 40
Program Memory Read Cycle

OSD-Input/Output Timing

Table 36

Parameter	Symbol	Limit Values Fixed DOT Clock f_{DOT} = 12 MHz		Unit
		min.	max.	
L-sandcastle time	t_{SCL}	15		µs
H-sandcastle time	t_{SCH}	3		µs
Horizontal offset	t_{HO}	t_{SCH}		µs
Pixel width	t_{DOT}	83		ns

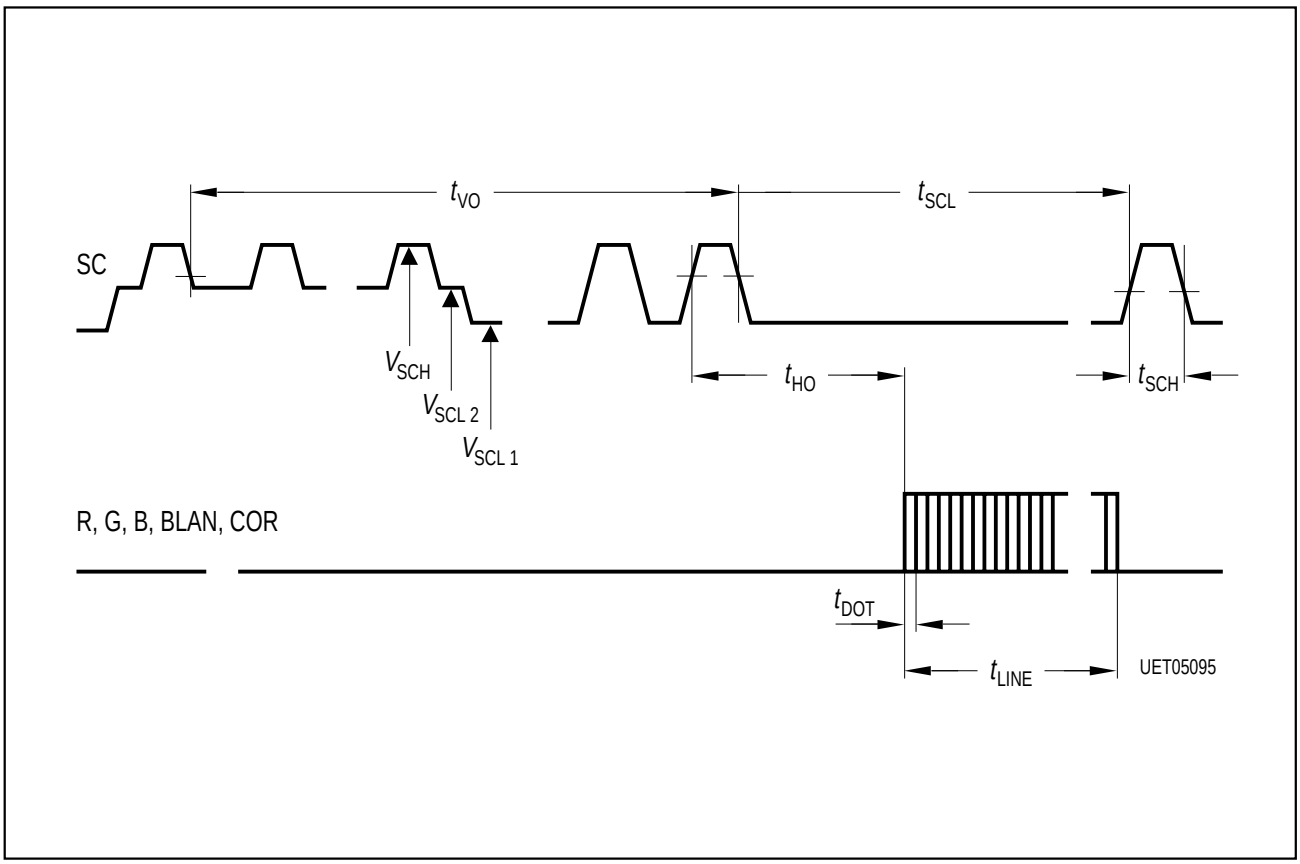


Figure 41
OSD-Input/Output Timing

Display-Generator-Timing

Table 37

T_A = 0 to 70 °C; V_{DD} = 5 V ± 10 %, V_{SS} = 0 V

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Hsync width	t_{HHHL}	2.8 / 1.4[1)]	–	µs
End of visible screen area to Hsync '1'	t_{VLHH}	0	–	ns
Start of visible screen area to Hsync '0'	t_{HLVH}	0	–	ns
Delay between Hsync and R/G/B/BLAN/COR-lines	t_{HHCL}	25	100	ns

[1)] default after reset is 2.8 µs; if bit 7 in SFR $0CD_H$ is set, the second value is valid

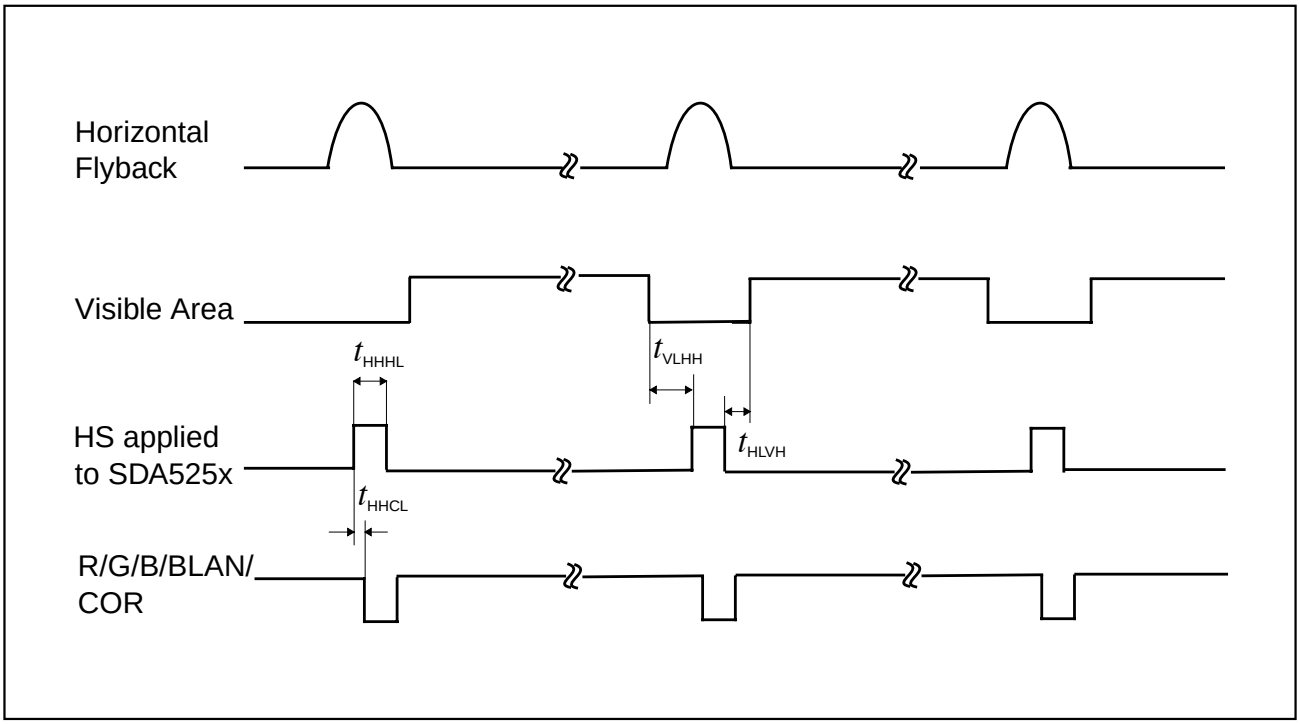


Figure 42
Horizontal Sync-Timing

AC-Testing Input, Output, Float Waveforms

AC testing inputs are driven at V_{DD} – 0.5 V for a logic ‘1’ and at 0.45 V for a logic ‘0’. Timing measurements are made at V_{IHmin} for a logic ‘1’ and at V_{IHmax} for a logic ‘0’. For timing purposes a port pin is no longer floating, when a 100 mV change from load voltage occurs.

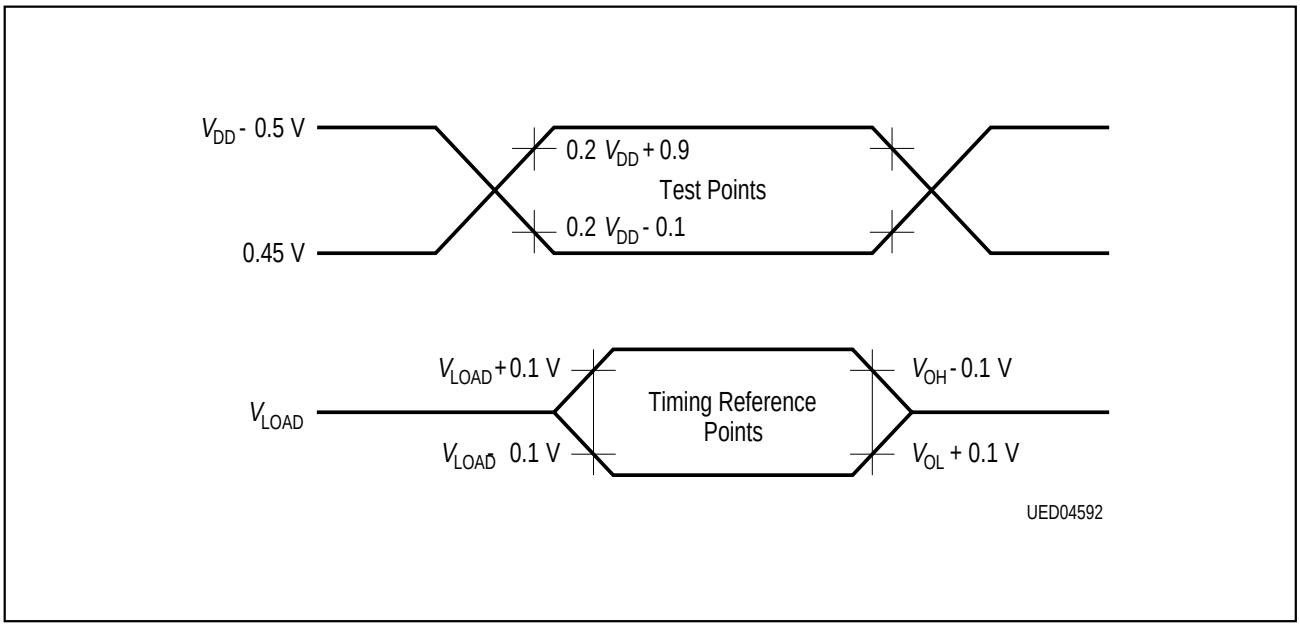


Figure 43
I/O-Waveform for AC-Tests

8 Applications

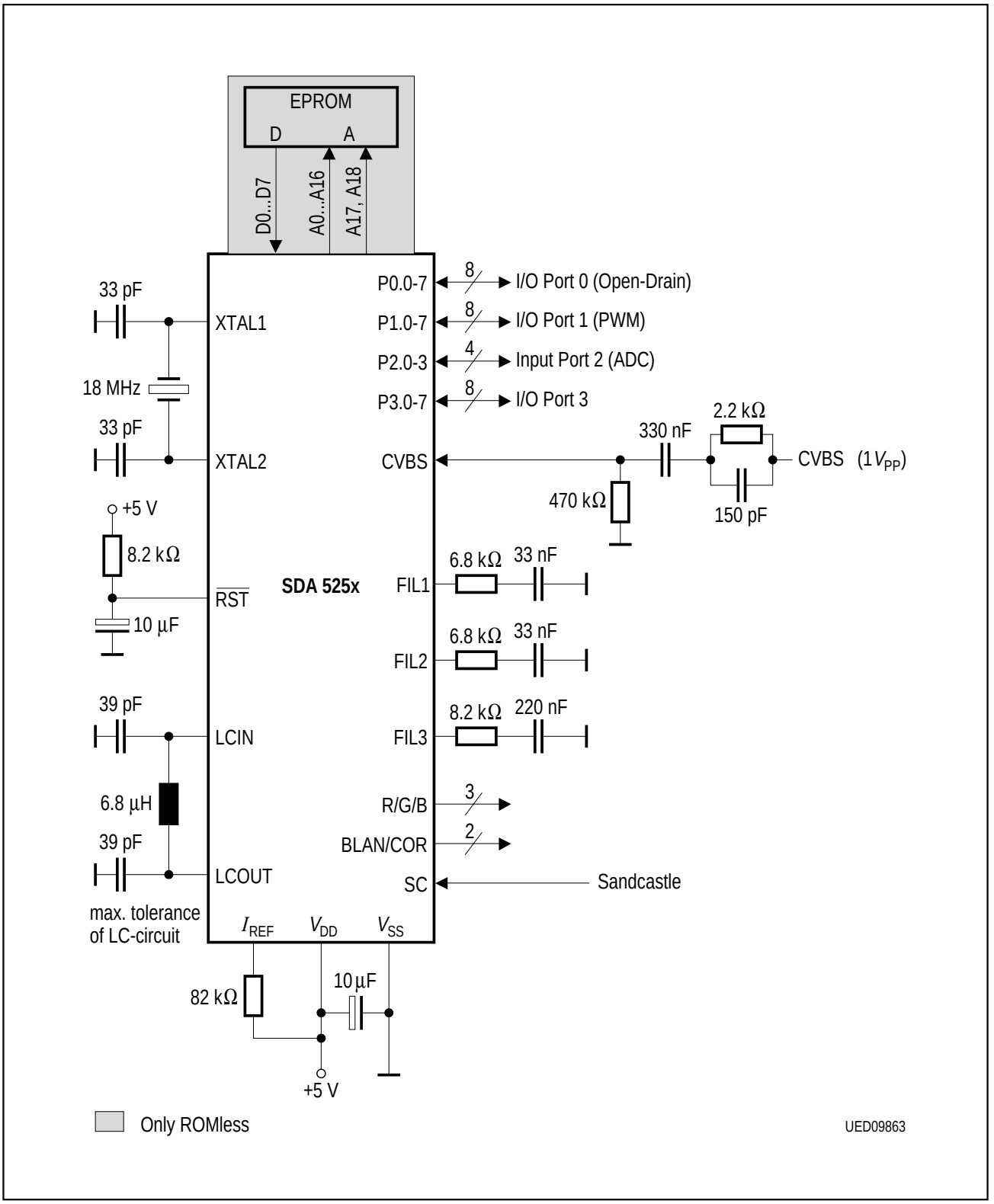
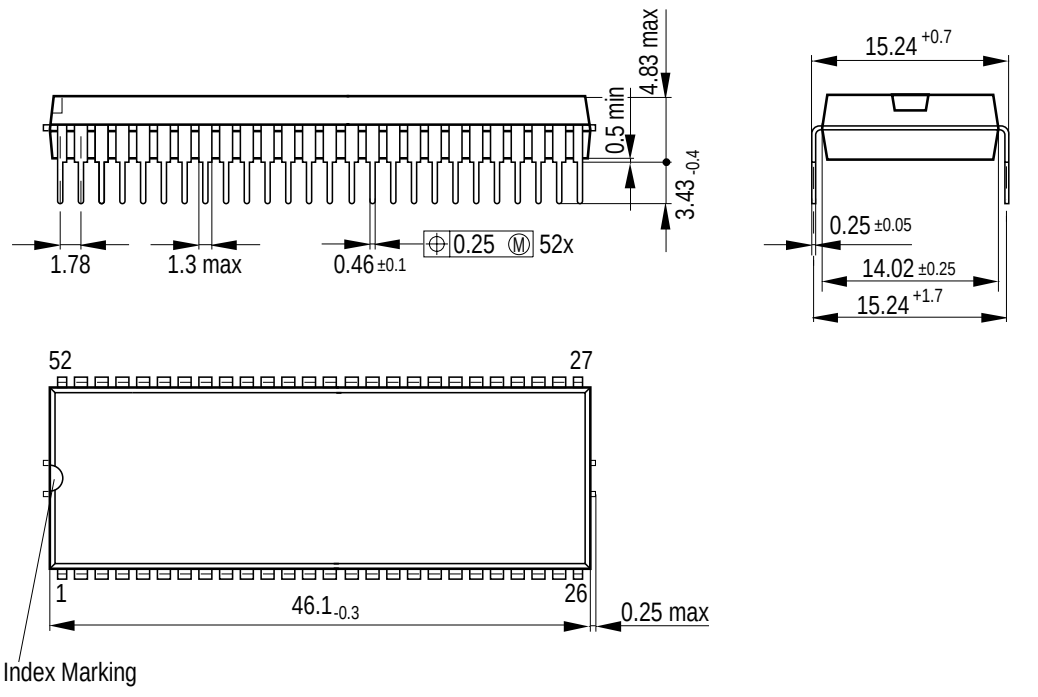


Figure 44
Application Circuit for 50 Hz Field Frequency

9 Package Outlines

P-SDIP-52-1
Plastic Shrink Dual In-line Package

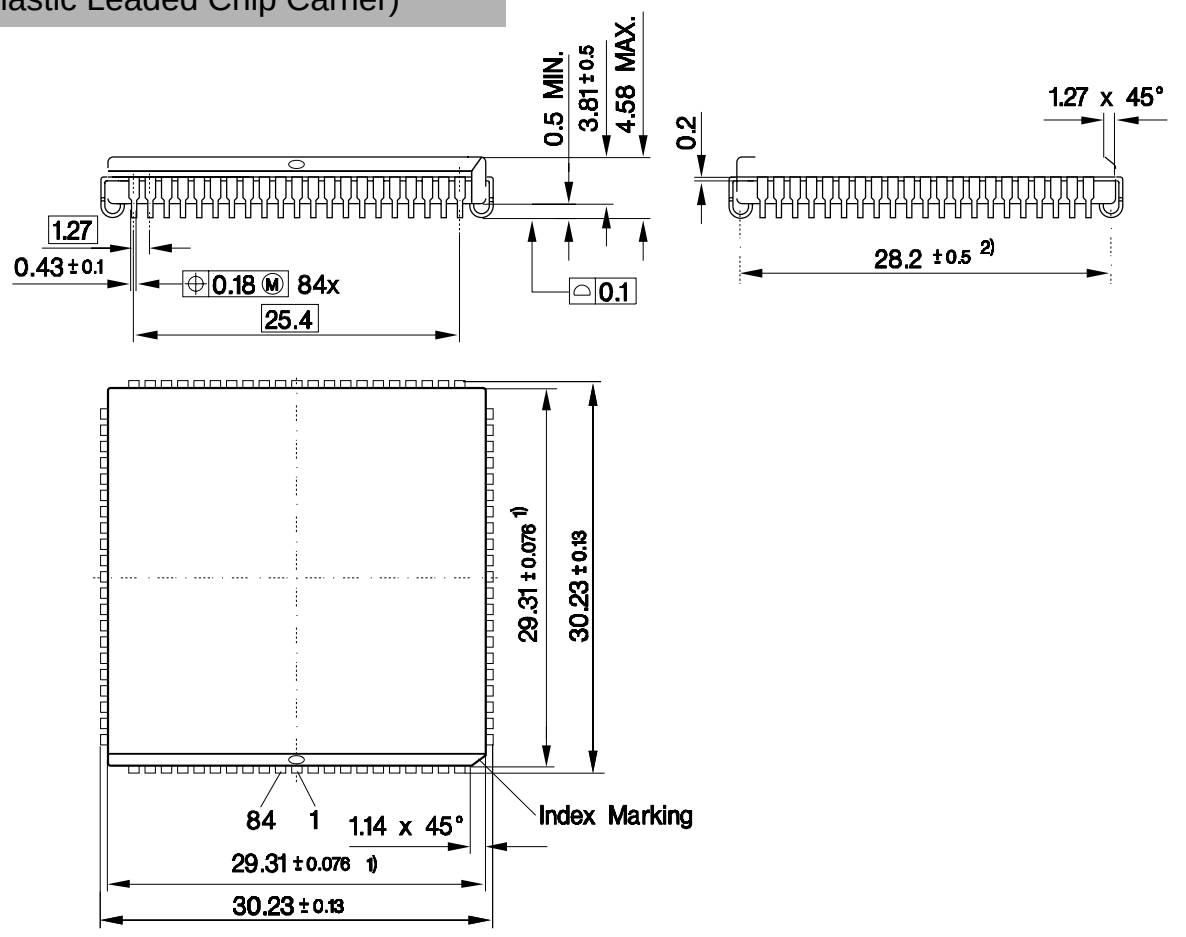


Sorts of Packing
Package outlines for tubes, trays etc. are contained in our Data Book “Package Information”.

SMD = Surface Mounted Device

Dimensions in mm

P-LCC-84-2
(Plastic Leaded Chip Carrier)



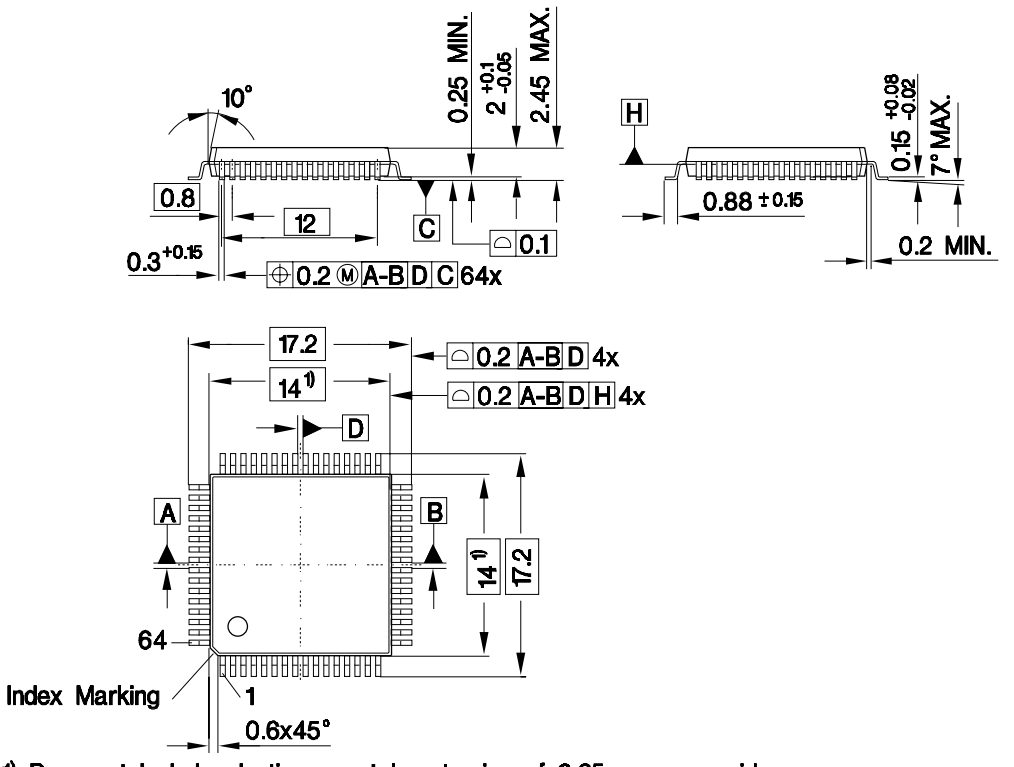
1) Does not include plastic or metal protrusion of 0.25 max. per side
2) Dimension from center to center

GPL05620

Sorts of Packing
Package outlines for tubes, trays etc. are contained in our Data Book “Package Information”.
SMD = Surface Mounted Device

Dimensions in mm

P-MQFP-64-1
(Plastic Metric Quad Flat Package)

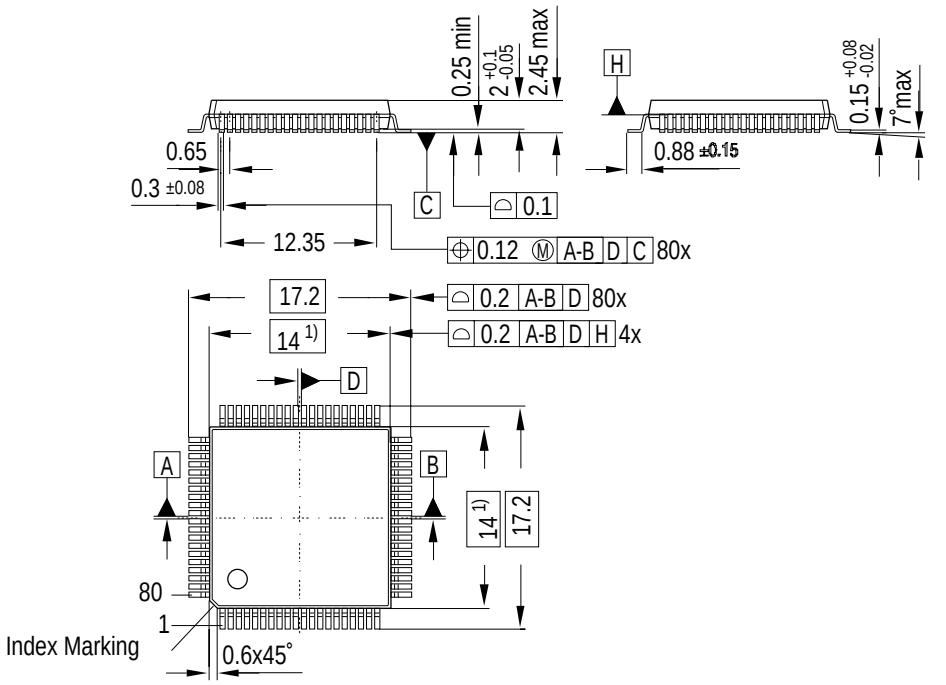


Sorts of Packing
Package outlines for tubes, trays etc. are contained in our Data Book “Package Information”.

SMD = Surface Mounted Device

Dimensions in mm

P-MQFP-80-1
(Plastic Metric Quad Flat Package)



1) Does not include plastic or metal protrusions of 0.25 max per side

GPM05249

Sorts of Packing
Package outlines for tubes, trays etc. are contained in our Data Book “Package Information”.

SMD = Surface Mounted Device

Dimensions in mm

10 Index