AN1155
APPLICATION NOTE

Connecting the ST10 Microcontroller to M29 Series Flash Memories

CONTENTS

INTRODUCTION

This application note describes specifically the connection of an M29F400B Flash memory to an ST10F163 microcontroller, although it can also be used as a reference for other Flash memory devices and other microcontrollers. The application note discusses the Bus Architectures of the Flash and the ST10. The additional features of the Flash, Reset/Block Temporary Unprotect and Ready/Busy Output are also discussed.

The information in this application note is useful to hardware engineers who are about to design a new circuit using a Flash memory.

ADVANTAGES OF FLASH

Flash memories can be used to store both code and data for microprocessors in the ST10 family. The ST10 can be configured to boot from the Flash and to execute the application software from it. Unlike EPROM, Flash can be used to store data; the ST10 can erase old, unwanted data and replace it. The application software can also be updated so that field upgrades can be performed without disassembling the product.

The ST10F163 contains 128Kbytes of internal Flash. The internal Flash can contain the program code that the microcontroller runs. However, the internal Flash cannot always be used for data storage because:

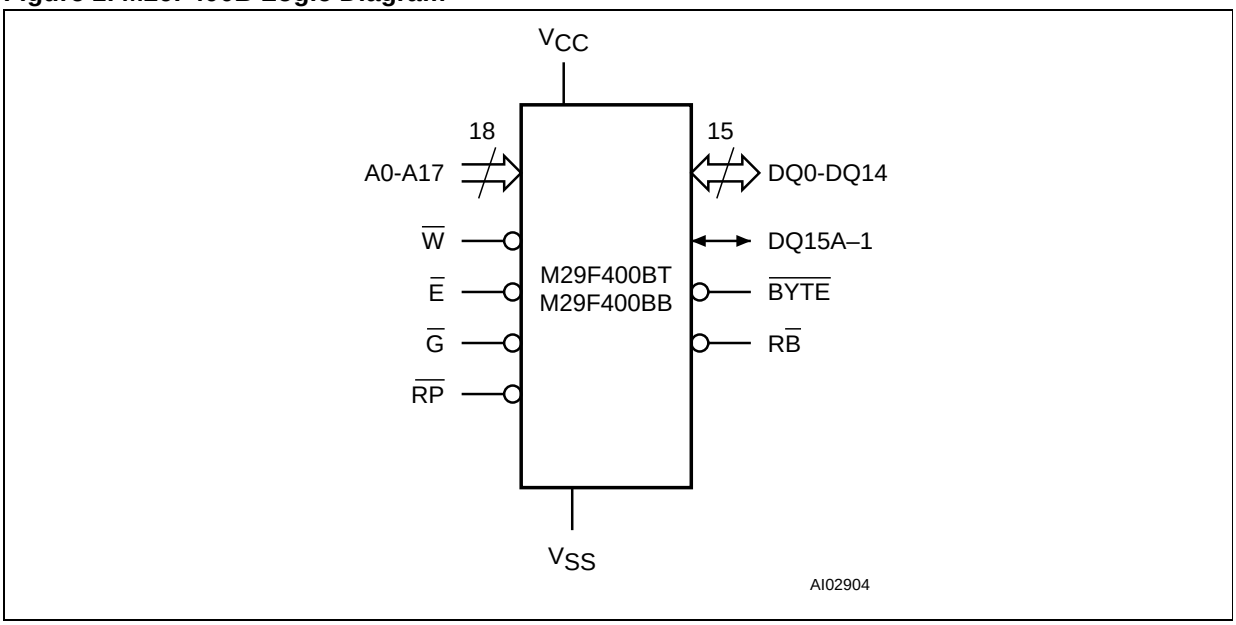
1. The internal Flash has a program/erase endurance of only 1000 cycles and it is therefore not suitable for frequent updates.
2. Programming and erasing the internal Flash requires 12V to be applied to the ST10 and a 12V supply is often not available.
3. Interrupts cannot be run from the internal Flash while it is programming or erasing. Hard real-time systems may not be able to tolerate the system becoming unavailable for 1 second or more while the internal Flash is erasing.

Putting an additional Flash memory on to an ST10F163 alleviates these problems. Extra space can be provided for code and data and the system can continue to be powered from a single 5V supply. Additionally the code can be run from the internal Flash while the external Flash is updated. Interrupts and high-priority tasks can run from the internal Flash while the external Flash is programming or erasing. Access to data in the external Flash is possible during erase operations using the Erase Suspend feature of the M29F400B.

FLASH BUS ARCHITECTURE

Take a look at the bus on the M29F400B, Figure 1 shows the Logic Diagram. The memory has separate Address and Data Buses ("de-multiplexed" in the ST10 terminology). The control lines are Chip Enable ($\overline{E}$), Output Enable ($\overline{G}$) and Write Enable ($\overline{W}$). Also, Ready/Busy Output ($R\overline{B}$) and Reset/Block Temporary Unprotect ($\overline{RP}$) are present. Finally there is the $\overline{BYTE}$ pin that selects 8-bit or 16-bit mode.

Figure 1. M29F400B Logic Diagram



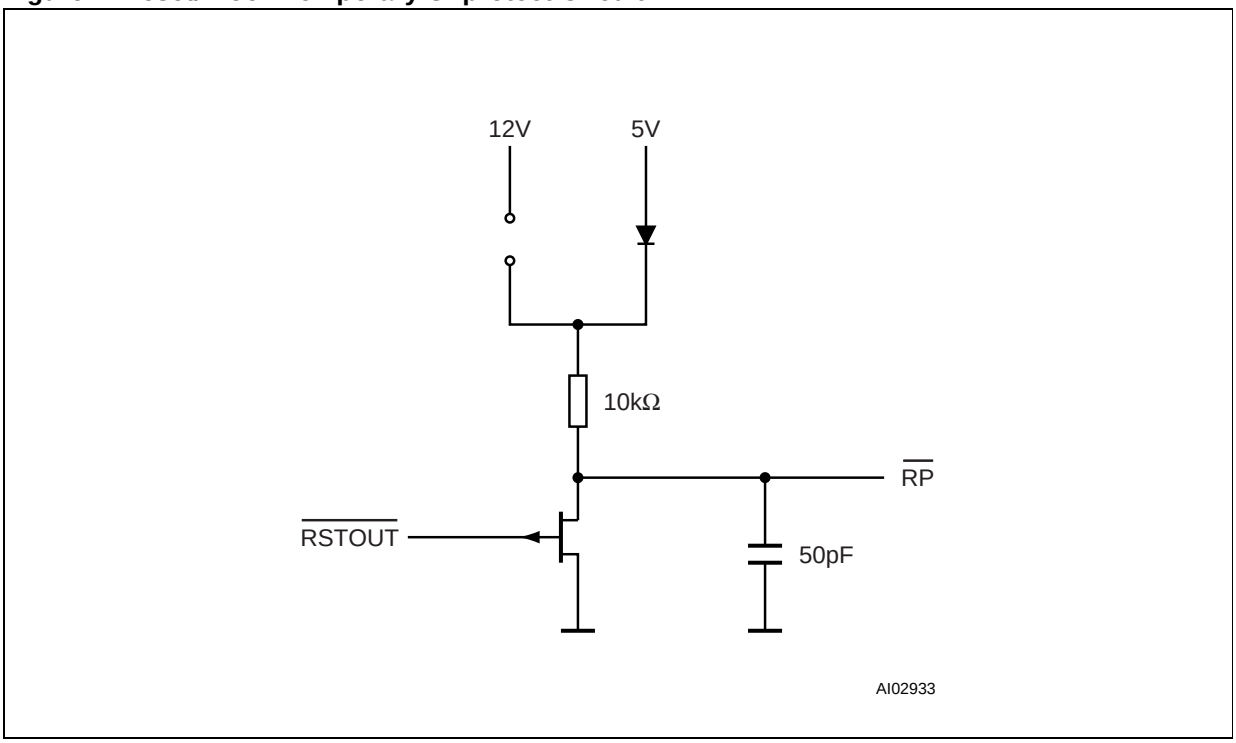
The M29F400B has been designed to allow $\overline{BYTE}$ to change between accesses, so 8-bit accesses and 16-bit accesses can be achieved in the same design. In practice it is far simpler to always use the memory in one mode; in order to swap modes additional logic is required to decode the DQ15A–1 pin, this would complicate the design, add cost and probably increase the wait-states required to access the memory.

In the example here 16-bit mode has been chosen. It is possible to read 8-bits in at a time, but it may be necessary to implement a TRAP on the ST10 in order to do this. All write accesses will write 16-bits at a time. In order to change one byte in the memory it will be necessary to write a word. Programming words and bytes takes the M29F400B the same amount of time; the internal charge pumps required for the program operation are word-wide, not byte-wide. Once the choice has been made to keep $\overline{BYTE}$ high, the special pin, DQ15A–1, can be treated like any other Data Input/Output pin. It forms part of the Data Bus, DQ0-DQ15. Note that the A0 address pin on the M29F400B specifies the address of a word, not of a byte; the address bus of the ST10F163 will need to be shifted compared to the address bus of the M29F400B in order to address the Flash correctly.

The Reset/Block Temporary Unprotect pin ($\overline{RP}$) accepts three states: Reset (V_{IL}), Not Reset (V_{IH}) and Block Temporary Unprotect (V_{ID}). Reset and Not Reset are the usual signals for a Reset line. The ST10F163 provides these two signals on its $\overline{RSTOUT}$ pin. The third state, Block Temporary Unprotect is used to temporarily unprotect blocks that have been specifically protected in the memory. Many applications do not protect any blocks and therefore connect the $\overline{RP}$ pin directly to the system Reset signal.

Figure 2 gives an example of how the connection between the ST10F163's $\overline{RSTOUT}$ pin and the M29F400B's $\overline{RP}$ pin can be made. The circuit makes use of a jumper to enable Block Temporary Unprotect. Many applications will provide the 12V from an external source, in which case the jumper can be replaced by a connector. The advantage with the circuit, as it stands, is that a reset from the ST10 will override Block Temporary Unprotect and cause the Flash to reset. Only four additional components are required.

Figure 2. Reset/Block Temporary Unprotect Circuit



Before the jumper is inserted, and when $\overline{\text{RSTOUT}}$ is High, V_{IH}, $\overline{\text{RP}}$ is connected to 5V through the 10kΩ resistor and the diode. The current required by $\overline{\text{RP}}$ is very low, in the order of 1µA at 5V. The voltage drop in the resistor and the diode at these currents will keep $\overline{\text{RP}}$ very close to 5V. When the jumper is fitted the diode ceases to conduct and $\overline{\text{RP}}$ rises to 12V as the capacitor charges. The time-constant of a 10kΩ resistor and a 50pF capacitor is 500ns, satisfying the t_{PHPHH} rise-time requirements of the M29F400B. During a Reset, $\overline{\text{RSTOUT}}$ is Low, V_{IL}, and the JFET is switched on, bringing $\overline{\text{RP}}$ close to ground. The current consumption during a Reset rises due to the current through the 10kΩ resistor.

Although the use of a jumper may not be the most elegant solution, it is a practical one because it maintains the security level offered by the Block Protection. There is little point in having the Block Temporary Unprotect pin under software control. The whole point of the Block Protection feature is to protect against software failure. Allowing the Block Temporary Unprotect feature to be under the control of software is nearly equivalent to not protecting the blocks in the first place.

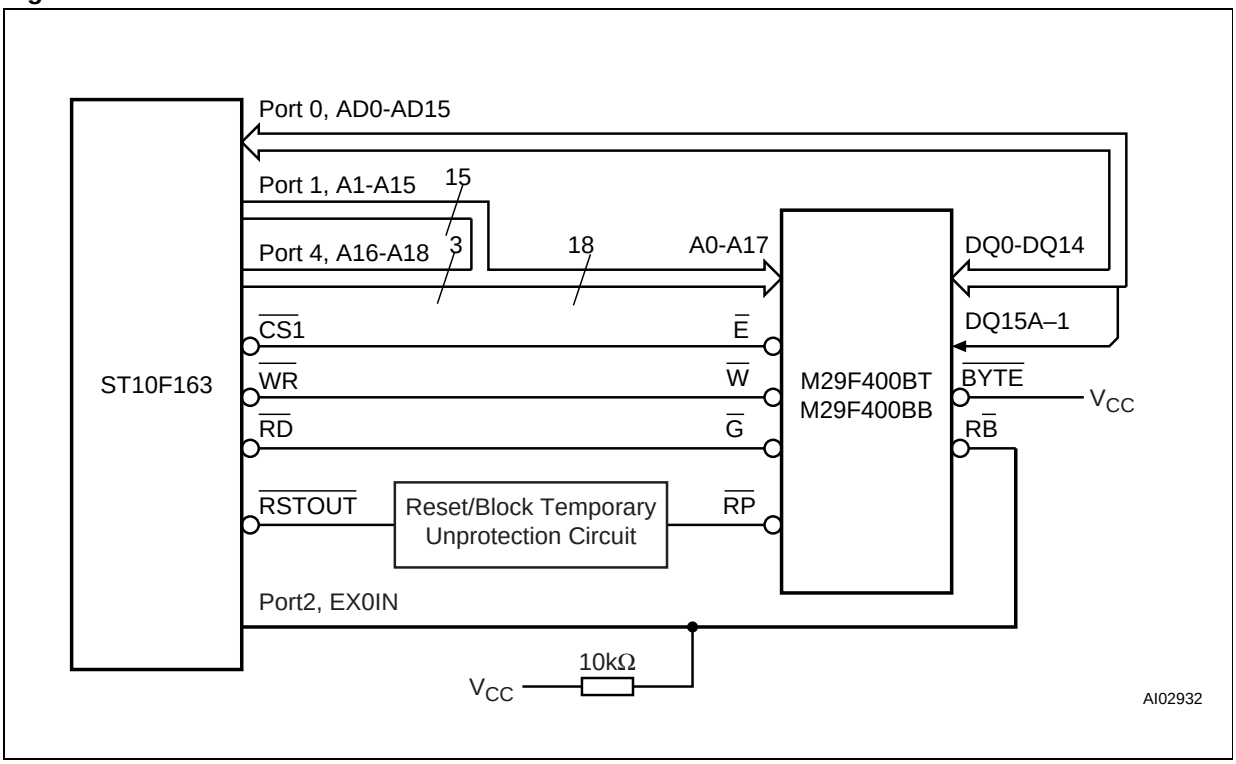
The Ready/Busy Output ($\text{R}\overline{\text{B}}$) provides a simple mechanism for determining if the Flash is busy or not. By connecting the Ready/Busy pin up to an interrupt line of the microcontroller it is possible to program the Flash under interrupt control, freeing the ST10 to perform other tasks while the Flash is programming. Otherwise the Flash must be polled in order to find out if program or erase operations have completed. Ready/Busy is an open drain output and, therefore, requires a pull-up resistor to bring the line up to V_{CC} when the Flash is ready.

One situation that the software must be aware of is the error situation. When a program or erase error occurs Ready/Busy will remain low until the error is cleared. A Timer, or polling algorithm, should be used to catch this situation and deal with it correctly.

ST10F163 BUS ARCHITECTURE

The ST10F163's bus architecture allows for both multiplexed buses and de-multiplexed buses. The simplest connection to the M29F400B is using a de-multiplexed bus. The Address Bus is on Port 1 (A0-A15) and Port 4 (A16-A23). The Data Bus is on Port 0 (D0-D15). The ST10F163 provides External Memory Write Strobe ($\overline{WR}$) and External Memory Read Strobe ($\overline{RD}$) signals. These are directly compatible with the Write Enable ($\overline{W}$) and Output Enable ($\overline{G}$) required by the M29F400B. There are also five Chip Selects ($\overline{CS0}$-$\overline{CS4}$). $\overline{CS1}$ has been used here to control accesses to the M29F400B's Chip Enable ($\overline{E}$), though any of the other four would work similarly. The connection between the ST10F163 and the M29F400B can be achieved without additional glue logic. Figure 3 shows the connection diagram.

Figure 3. Connection between the ST10F163 and the M29F400B



Note that the connection between the Flash memory address bus and the ST10F163 address bus is shifted. A0 on the ST10F163 is not connected to the Flash memory as this bit of the address bus selects between the LSB and the MSB. The Flash memory only outputs word-width values, each increment in the Flash's address space selects the next word. To make the two address buses compatible A1 on the ST10F163 should be connected to A0 on the Flash memory; A2 on the ST10F163 should connect to A1 on the Flash memory; etc. The highest ST10F163 memory address bit used is A18, which connects to A17 on the Flash. The higher address bits are not required (for example for address decoding) since the address decode logic is internal to the ST10F163 and $\overline{CS1}$ is used to enable the Flash.

TIMING REQUIREMENTS

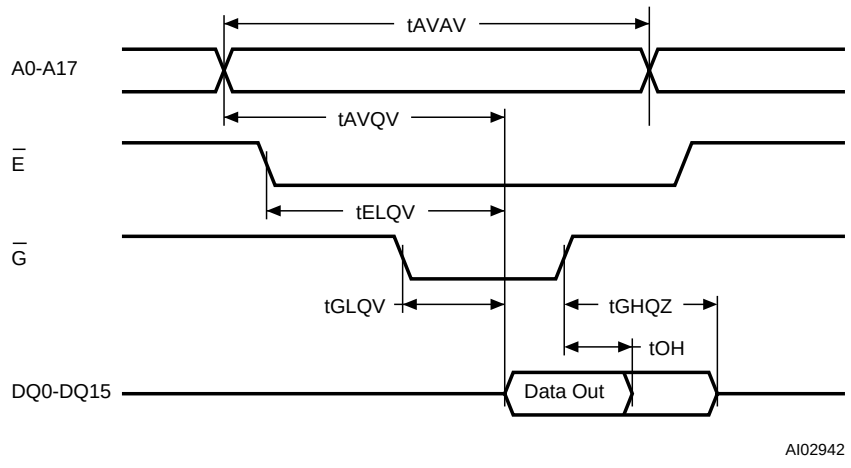
The ST10F163 bus is very adaptable, allowing for changes to wait-states, Read/Write delays, Chip Select delays and Tri-State delays. The timings are shown with the ST10 configured for Unlatched CSx, Read/Write delay, Normal ALE and no Tri-State delay. The wait-states are determined by t_A (address set-up wait-states), t_C (cycle wait-states) and t_F (tri-state wait-states). In all cases t_A = 0ns and t_F = 0ns. One wait-state with f_{CPU} = 25MHz gives t_C = 40ns, otherwise t_C = 0ns (zero wait-states).

Table 1 shows the principal read timing requirements of the M29F400B and the read timings generated by the ST10F163 (inapplicable timings have been left out). Figure 4 shows the principal read timing waveforms. Table 2 shows the principal write timings and Figure 5 the principal write timing waveforms.

Table 1. Principal Read Timing Requirements

M29F400B				ST10F163		
Symbol	**45**	**55**	**70**	**f_{CPU} = 25MHz t_C = 0ns**	**f_{CPU} = 25MHz t_C = 40ns**	**f_{CPU} = 20MHz t_C = 0ns**
t_{AVAV}	45	55	70	80	120	100
t_{AVQV}	45	55	70	50	90	70
t_{ELQV}	45	55	70	50	90	70
t_{GLQV}	25	30	30	20	60	30
t_{GHQZ}	15	18	20	26	26	36
t_{OH}	0	0	0	0	0	0

Figure 4. Principal Read Timing Waveforms

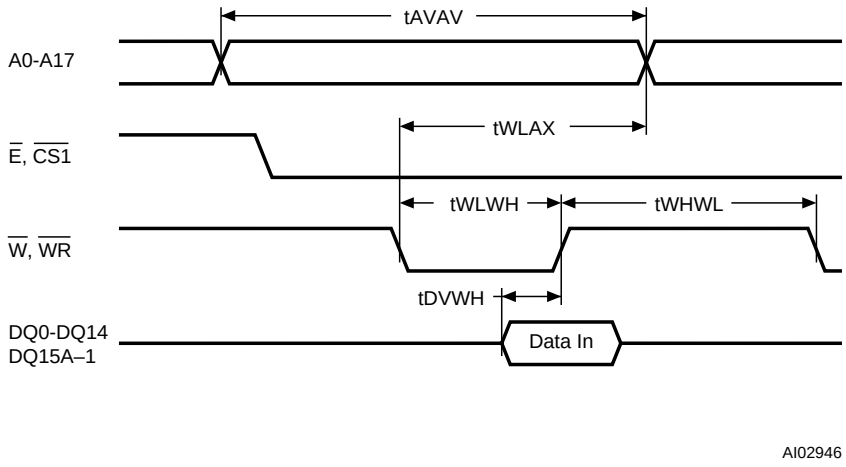


The 45ns access time M29F400B cannot satisfy all of the timing requirements of the ST10F163 at 25MHz with zero wait-states: t_{GLQV}, t_{WLWH}, t_{DVWH} and t_{WLAX} all miss the ST10F163 timing requirements. Introducing one wait-state allows all of the timing requirements to be comfortably satisfied by the 70ns part. Reducing the ST10F163 clock speed to 20MHz also allows all of the timing requirements to be satisfied by the 55ns part, but without any margin.

Table 2. Write Timing Requirements, Write Enable Controlled

M29F400B				ST10F163		
Symbol	**45**	**55**	**70**	**f_{CPU} = 25MHz t_C = 0ns**	**f_{CPU} = 25MHz t_C = 40ns**	**f_{CPU} = 20MHz t_C = 0ns**
t_{AVAV}	45	55	70	80	120	100
t_{WLWH}	40	40	45	30	70	40
t_{DVWH}	25	25	30	20	60	30
t_{WHWL}	20	20	20	40	40	50
t_{WLAX}	40	40	45	30	70	40

Figure 5. Principal Write Timing Waveforms



CONCLUSION

The M29F400B, and other Flash memory parts from STMicroelectronics, can be easily connected to the ST10F163 using no glue logic. Zero or one wait-state solutions can be realized. The Block Temporary Unprotect feature can be designed in easily and the software overhead for programming the Flash can be reduced by connecting the Ready/Busy Output pin to an interrupt line.

If you have any questions or suggestion concerning the matters raised in this document please send them to the following electronic mail address:

ask.memory@st.com (for general enquiries)

Please remember to include your name, company, location, telephone number and fax number.