UM0470
User manual

STM8 SWIM communication protocol and debug module

Introduction

This manual has been written for developers who need to build programming, testing or debugging tools for the STM8 microcontroller family. It explains the debug architecture of the STM8 core.

The STM8 debug system consists of 2 modules.

DM - Debug module

SWIM - Single wire interface module

Related documentation

- STM8 Flash programming reference manual (PM0047)

Contents

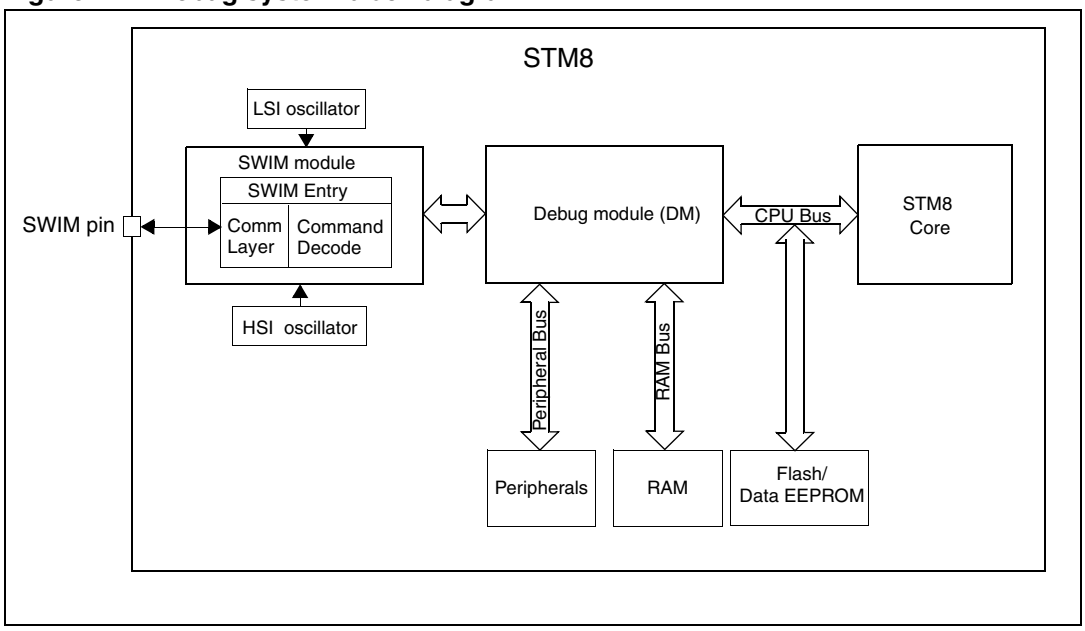
1 Debug system overview

The STM8 debug system interface allows a debugging or programming tool to be connected to the MCU through a single wire bidirectional communication based on open-drain line.

It provides non-intrusive read/write access to RAM and peripherals during program execution

The block diagram is shown in *Figure 1*.

Figure 1. Debug system block diagram



The debug module uses the two internal clock sources present in the device, the LSI Low Speed Internal clock (usually in the range 30 kHz-200 kHz, depending on the product) one and the HSI High Speed Internal clock (usually in the range 10 MHz to 25 MHz, depending on the device). The clocks are automatically started when necessary.

2 Communication layer

The SWIM module is a single wire interface based on asynchronous, high sink (8 mA), open-drain, bidirectional communication.

While the CPU is running, the SWIM module allows non-intrusive read/write accesses to be performed on-the-fly to the RAM and peripheral registers, for debug purposes.

In addition, while the CPU is stalled, the SWIM module allows read/write accesses to be performed to any other part of the MCU's memory space (Data EEPROM and program memory).

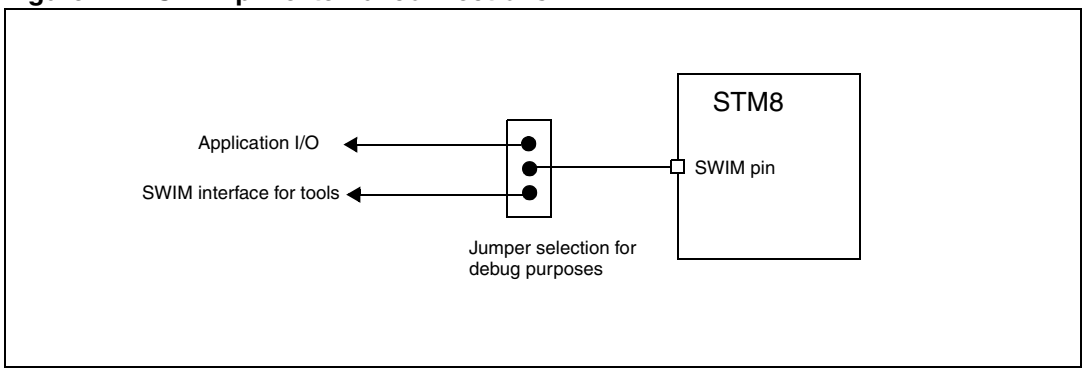
CPU registers (A, X, Y, CC, SP) can also be accessed. These registers are mapped in memory and can be accessed in the same way as other memory addresses.

- Register, peripherals and memory can be accessed only when the SWIM_DM bit is set.
- When the system is in HALT, WFI or readout protection mode, the NO_ACCESS flag in the SWIM_CSR register is set. In this case, it is forbidden to perform any accesses because parts of the device may not be clocked and a read access could return garbage or a write access might not succeed.

The SWIM module can perform a MCU device software reset.

The SWIM pin can also be used by the MCU target application as a standard I/O port with some restrictions if you also want to use it for debug. The safest way is to provide a strap option on the application PCB.

Figure 2. SWIM pin external connections



3 Single wire interface module (SWIM)

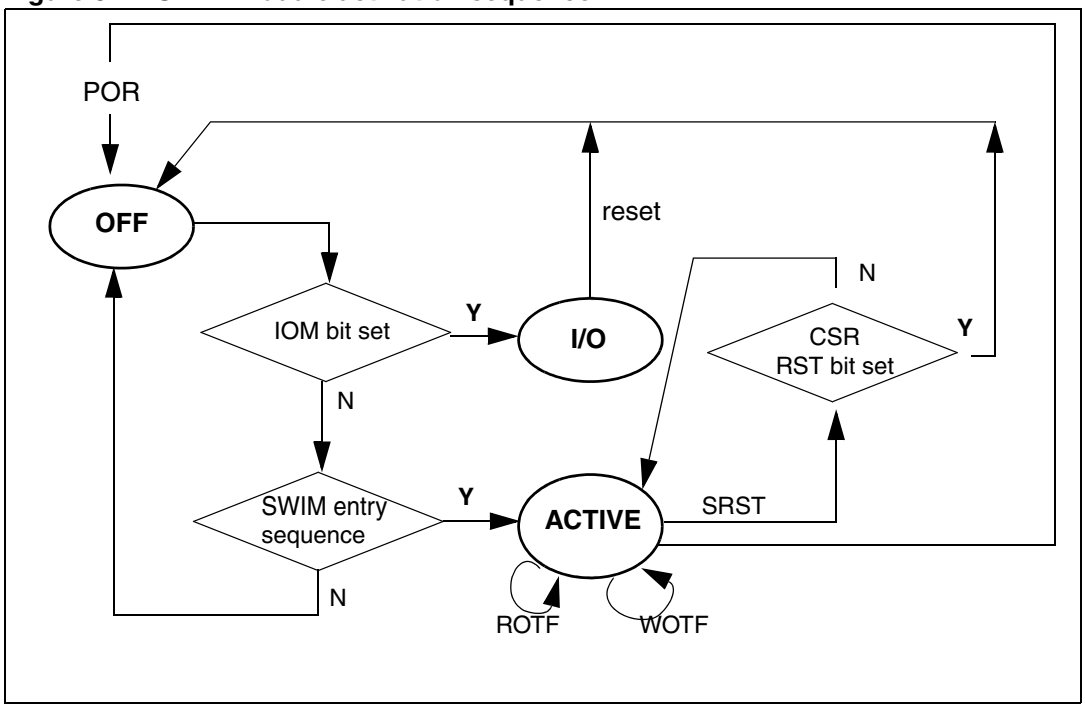
3.1 Operating modes

After a Power-On Reset (powering of the device) the SWIM module is reset and enters in its OFF mode.

1. **OFF**: In this mode the SWIM pin must not be used as an I/O by the application. It is waiting for the SWIM entry sequence or to be switched to I/O mode by the application software.
2. **I/O**: This state is entered by the software application by setting the IOM bit in the core configuration register (MCR). In this state, the user application can use the SWIM pin as a standard I/O pin, the only drawback is that there is no way to debug the functionality of this pin with the built-in debug capabilities.
   In case of a reset, the SWIM goes back to OFF mode.
3. **ACTIVE**: This mode is entered when a specific sequence is detected on the SWIM pin while in OFF state. In this state, the SWIM pin is used by the host tool to control the STM8 with 3 commands. (SRST System Reset, ROTF Read On The Fly, WOTF Write On The Fly)

Note: Please note that the SWIM can be set Active and communicate while the device is in RESET state (NRST pin forced low)

Figure 3. SWIM module activation sequence



3.2 SWIM entry sequence

After a POR, and as long as the SWIM module is in OFF mode, the SWIM pin is sampled for entry sequence detection. In order to do this, the internal low speed RC clock is automatically turned ON after POR and remains forced ON as long as the SWIM is in OFF mode.

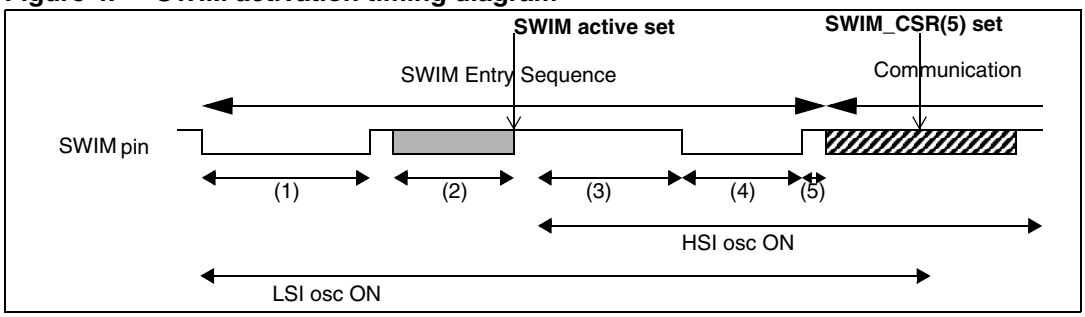
If the register which forces the SWIM module in I/O mode is written before the entry sequence is finalized, the SWIM module enters I/O mode. Once the SWIM module is ACTIVE, writing this bit has no influence on communication and the SWIM interface remains in ACTIVE mode.

If an application uses the SWIM pin as standard I/O, it puts the SWIM interface in I/O mode in the initialization section of the software code (typically, this is performed just after the reset). However, even in this case, it is still possible to put the SWIM interface in ACTIVE mode by forcing the RESET pin to 0 and keep it low for the duration of the SWIM entry sequence.

As long as the SWIM module is in OFF mode, the SWIM entry sequence is detected at any moment, during reset or when the application is running.

If both the SWIM pin and the reset pin are multiplexed with I/Os, the way to enter SWIM ACTIVE state is to power down the MCU device, power up and to maintain the reset until the end of the SWIM entry sequence.

Figure 4. SWIM activation timing diagram

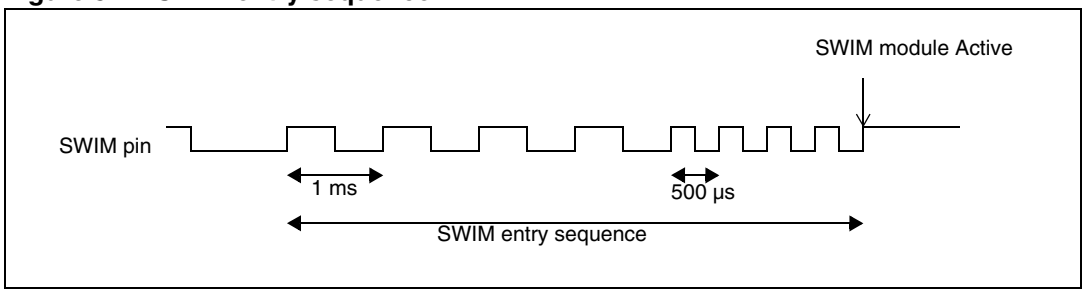


SWIM activation is shown on *Figure 4* and each segment on the diagram is described below.

1. To make the SWIM active, the SWIM pin must be forced low during a period of several µs.
2. After this first pulse at 0, the SWIM module will detect a **specific sequence** to guarantee robustness in the SWIM active state entry. The SWIM entry sequence is: 4 pulses at 1 kHz followed by 4 pulses at 2 kHz. The frequency ratio is detected and allows SWIM entry. The ratio can be easily detected whatever internal RC frequency. The waveform of the entry sequence is shown *Figure 5*.
3. After the entry sequence, the SWIM enters in SWIM active state, and the HSI oscillator is automatically turned ON.
4. In most cases, SWIM entry is done when the RESET line is active (low). In order to perform device calibration, the tool must release the RESET line and calibrate the clock by sending a “SWIM communication reset” command after the **option bytes loading** phase, in order to reach a precision of +/- 4%. The duration of this phase is 64 x HSI clock periods.
5. After this delay, the SWIM module sends a **synchronization** frame to the host.

   Synchronization frame description: A synchronization frame of 128 x HSI clock periods with the SWIM line at 0 is sent out by the MCU device to allow for the measurement of the RC by the debug host. An advanced debug host can re-calibrate its clock to adapt to the frequency of Internal RC.
6. Before starting a SWIM communication, the SWIM line must be released at 1 to guarantee that the SWIM module is ready for communication (at least 300 ns).

Figure 5. SWIM entry sequence



3.3 Bit format

The bit format is a Return-To-Zero format, which allows synchronization of every bit.

Two communication speeds are available. At SWIM activation, the low speed is selected. The high speed is selected by setting the HS bit in the SWIM_CSR register with the SWIM protocol.

When entering SWIM mode during the RESET phase, it is possible that the option bytes have not yet been loaded from non volatile memory to their respective registers.

Option byte loading is triggered by any internal or external reset.

In order to ensure proper system behavior, the HS bit should not be set until the option byte loading is finished. At the end of the option byte loading, the OBL bit in the SWIM_CSR is set by hardware.

3.3.1 High speed bit format

1 bit is generated with ten HSI oscillator pulses.

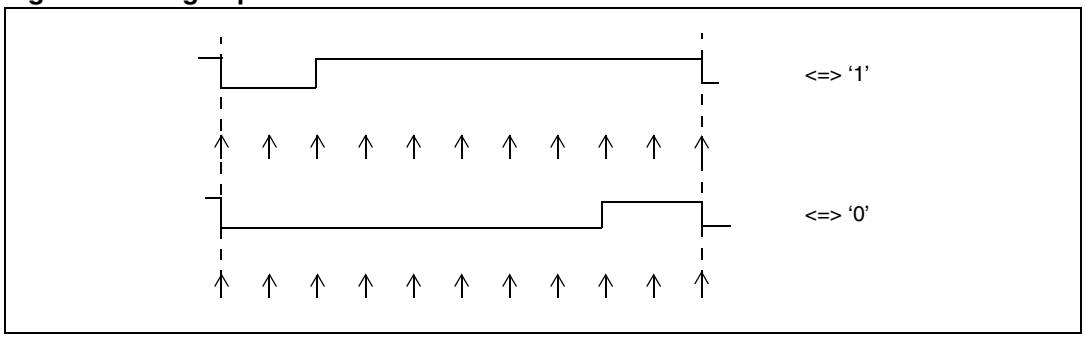
The bit format is:

- 2 pulses at ‘0’ followed by 8 pulses at ‘1’ for ‘1’ value.
- 8 pulses at ‘0’ followed by 2 pulses at ‘1’ for ‘0’ value.

When the SWIM module receives a data packet, it will decode:

- ‘1’ when the number of consecutive samples at ‘0’ is less or equal to 4.
- ‘0’ when the number of consecutive samples at ‘0’ is greater or equal to 5.

Figure 6. High speed bit format



3.3.2 Low speed bit format

1 bit is generated with twenty-two HSI oscillator pulses.

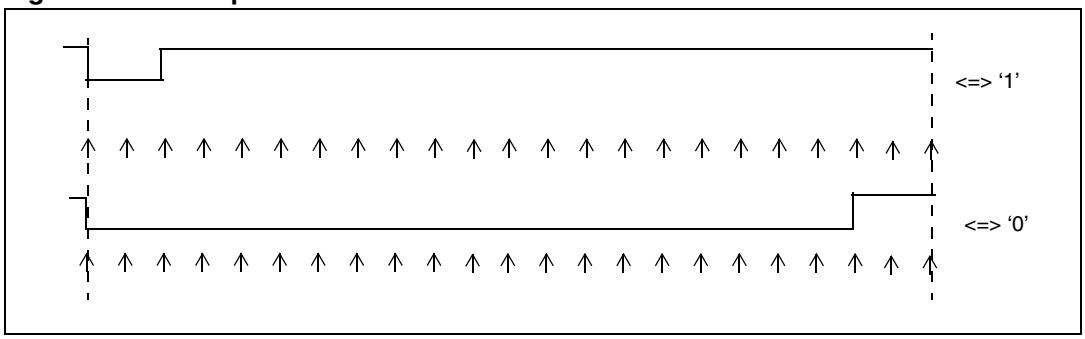
The bit format is:

- 2 pulses at ‘0’ followed by 20 pulses at ‘1’ for ‘1’ value.
- 20 pulses at ‘0’ followed by 2 pulses at ‘1’ for ‘0’ value.

When the SWIM module receives a data packet, it will decode:

- ‘1’ when the number of consecutive samples at ‘0’ is less or equal to 8.
- ‘0’ when the number of consecutive samples at ‘0’ is greater or equal to 9.

Figure 7. Low speed bit format

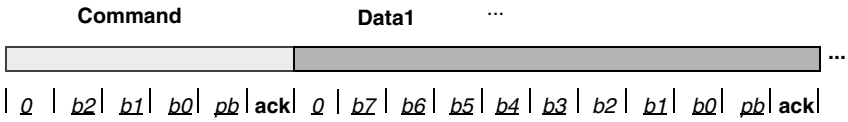


3.4 SWIM communication protocol

When in ACTIVE mode, communication can be initiated by host or device. Each byte or command is preceded by a 1-bit header in order to arbitrate if both host and device initiate the communication at the same time.

The host header is '0' in order to have the priority over the device in case of arbitration, due to open-drain capability. The host can start the transfer only if there is no transfer on-going.

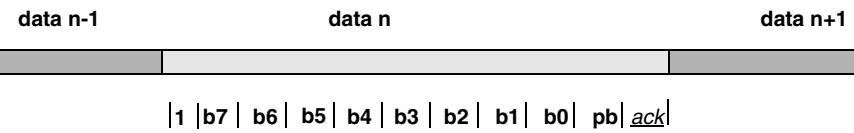
Figure 8. .Command Format (Host -> Target)



Each command sent by the host is made of:

- 1 command (ROTF, WOTF or SWRST) made of
  - Header: 1 bit at '0'
  - b2-b0: 3-bit command
  - pb: parity bit: XOR between all b(i)
  - ack:acknowledge (1 bit at '1'). The receiver must send the not-acknowledge value if it has detected a parity error (NACK: not acknowledge = 1 bit at '0'), or it is not yet ready.
- and optionally several data packets (in case of WOTF) made of:
  - Header: 1 bit at '0'
  - b7-b0: 8-bit data
  - pb: parity bit sent after data. XOR between all b(i)
  - ack: acknowledge

Figure 9. Data Format (Target -> Host)



Each data frame is made of:

- Header: 1 bit at '1'
- b7-b0: 8-bit data
- pb: parity bit sent after data
- ack: acknowledge

3.5 SWIM commands

The Host can send a command when the line is idle or after each data byte from device. After sending the command, the host releases the line. When the SWIM is ready to answer to the command, it initiates the transfer. If a new command from the host occurs while a command is pending in SWIM, the pending command is cancelled and the new command is decoded, except in case of WOTF.

Three commands are available. They are listed in *Table 1*.

Table 1. SWIM command summary

Command	Binary Code
SRST	000
ROTF	001
WOTF	010
Reserved for future use	011 1xx

3.5.1 SRST: system reset

Format: 1 command from Host to Target

SRST

Parameters:

None.

SRST command generates a system reset only if SWIM_CSR/SWIM_DM bit is set.

3.5.2 ROTF: read on the fly

Format: 1 command followed by the number of bytes to be read followed by the address on three bytes.

ROTF	N	@E	@H	@L		D[@]		D[@+N]

Parameters:

N The 8 bits are the number of bytes to read (from 1 to 255)

@E/H/L: This is the 24-bit address to be accessed.

D[...]: These are the data bytes read from the memory space

If the host sends a NACK to a data byte, the device will send the same byte again.

If SWIM_DM bit is cleared, ROTF can only be done on SWIM internal registers.

3.5.3 WOTF: write on the fly

1 command followed by the number of bytes to be written followed by the address on three bytes.

WOTF	N	@E	@H	@L	D[@]	D[@+N]

Parameters:

N	The 8 bits are the number of bytes to write (from 1 to 255)
@E/H/L:	This is the 24-bit address to be accessed.
D[...]:	These are the data bytes to write in the memory space

If a byte D [i] has not been written when the following byte D [i+1] arrives, D [i+1] will be followed by a NACK. In this case the Host must send D [i+1] again until it is acknowledged.

For the last byte, if it is not yet written when a new command occurs, the new command will receive a NACK and will not be taken into account.

If SWIM_DM bit is cleared, WOTF can only be done on SWIM internal registers.

3.6 SWIM communication reset

In case of a problem during communication, the host can reset the communication and the on-going command by sending 128 x HSI clock periods low on the SWIM pin. If the SWIM logic detects that the SWIM pin is low for more than 64 x HSI clock periods, it will reset the communication state machine and will switch the SWIM to low-speed mode (SWIM_CSR.HS <- 0). This is to allow for variation in the frequency of the internal RC oscillator.

In response to this communication reset, SWIM will send the synchronization frame which is 128x HSI oscillator periods low on DBG pin.

3.7 CPU register access

The CPU registers are mapped in the STM8 memory, and they can be read or written directly using the ROTF and WOTF SWIM commands. Write operations to the CPU registers are committed only when the CPU is stalled.

To flush the instruction decode phase, you must set the FLUSH bit in the *DM control/status register 2 (DM_CSR2)* after writing a new value in the Program Counter (PCE, PCH, PCL).

Table 2. CPU register memory mapping in STM8

CPU register	Memory location
A	7F00h
PCE	7F01h
PCH	7F02h
PCL	7F03h
XH	7F04h
XL	7F05h
YH	7F06h
YL	7F07h
SPH	7F08h
SPL	7F09h
CC	7F0Ah

3.8 SWIM communication in Halt mode

To maintain the communication link with the debug host, the HSI oscillator remains on when the MCU enters Halt mode. This means that halt mode power consumption measurements have no meaning when the SWIM module is active.

The NO_ACCESS bit in the SWIM_CSR register is set when the system is in HALT, WFI or readout protection mode. This means the bus is not accessible in this case.

The OSCOFF bit in the SWIM_CSR register is used to switch off the oscillator. In this case, debug control is lost as long as the device is in Halt mode and the SWIM pin is high. The only way to recover the debug control is to induce a falling edge on SWIM pin: this will re-enable the HSI oscillator.

3.9 Physical layer

During the communication, the SWIM pin will be in pseudo-open drain configuration. The SWIM pin in the device is capable of sinking 8 mA when it drives the line to 0. The external pull-up on the SWIM line should be sized in such a way that the maximum rise time t_r of the SWIM line should be less than 1 sampling period of the bit (which is 100 ns +/- 4 %).

Figure 10. Timings on SWIM pin

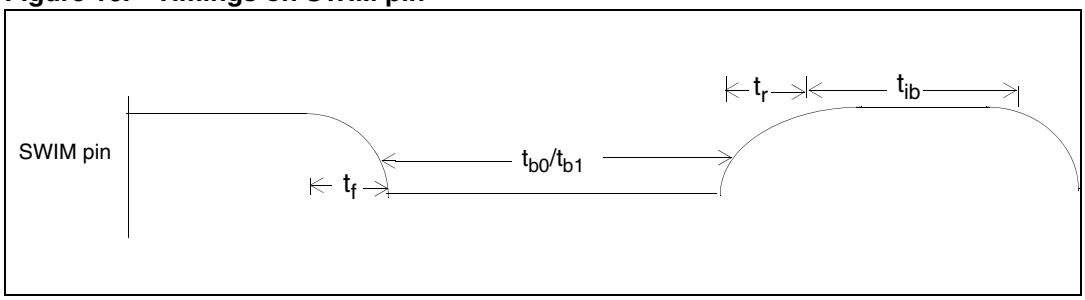


Table 3. SWIM pin characteristics

Parameter	Symbol	Generic formula	Timings for HSI = 10 MHz LSI = 32 to 64 kHz	
			Min	Max
Fall time on SWIM pin	t_f	TBD		50 ns
Rise time on SWIM pin	t_r	TBD		96 ns
Inter-bit time (The time which SWIM pin stays high between 2 bits)	t_{ib}	TBD	>0	
Inter-frame time (Time between end of a frame and the next one)	t_{if}	TBD	0	
Low time for a bit at 0	High speed: t_{b0}	TBD	768 ns	832 ns
	Low speed: t_{b0}	TBD	1.6 µs	2.4 µs
Low time for a bit at 1 (High Speed)	High speed: Tb1	TBD	192 ns	208 ns
	Low speed: Tb1	TBD	150 ns	250 ns
Injected current on SWIM pin		TBD		8 mA

3.10 STM8 SWIM registers

3.10.1 SWIM control status register (SWIM_CSR)

Address: 7F80h

Reset value: 00h

This register is reset only by a power on reset or by SWIM SRST command if the RST bit =1 in the SWIM_CSR register.

7	6	5	4	3	2	1	0
SAFE_MASK	NO_ACCESS	SWIM_DM	HS	OSCOFF	RST	HSIT	PRI
rw	r	rw	rw	rw	rw	r	rw

Bit 7	**SAFE_MASK**: *Mask internal RESET sources* This bit can be read or written through SWIM only. It cannot be accessed through the STM8 bus. It includes the Watchdog reset. 0: Internal Reset sources are not masked 1: Internal reset sources are masked
Bit 6	**NO_ACCESS**: *Bus not accessible* This bit can be read through SWIM only, to determine the bus is accessible or not. It is set automatically if the device is in HALT, WFI or readout protection mode. 0: Bus is accessible 1: Bus is not accessible **Caution:** Depending on the SWIM module revision, in some devices, the NO_ACCESS bit indicates only that the device is in HALT mode.
Bit 5	**SWIM_DM**: *SWIM for Debug Module* This bit can be read or written to 1 through SWIM only. It cannot be accessed through the STM8 bus. 0: The SWIM module can access only SWIM_CSR register. SWIM reset command has no effect 1: The whole memory range can be accessed with ROTF and WOTF commands. The SRST command generates a Reset
Bit 4	**HS**: *High Speed* This bit can be read or written through SWIM only. It cannot be accessed through STM8 bus. 0: Low speed bit format 1: High speed bit format The speed change occurs when the communication is IDLE. It is reset by the SWIM communication reset condition as described in *Section 3.6*.
Bit 3	**OSCOFF**: *Oscillators Off control bit* This bit can be read or written through SWIM only. It cannot be accessed through STM8 bus. 0: HSI oscillator remains ON in halt mode 1: HSI oscillator is not requested ON in Halt mode

Bit 2	**RST**: *SWIM Reset Control Bit* This bit can be read or written through SWIM only. It cannot be accessed through STM8 bus. 0: SWIM is not reset when a SRST command occurs. 1: SWIM is reset when a SRST command occurs. SWIM will re-enter OFF mode.
Bit 1	**HSIT**: *High Speed Internal Clock is trimmed* This bit is read only through SWIM only. It cannot be accessed through STM8 bus. It is set when the HSIT bit is set in the core configuration register and reset by an external reset. 0: High Speed Internal Clock is not trimmed, SWIM must remain in low speed mode. 1: High Speed Internal Clock is trimmed, SWIM high speed mode is allowed.
Bit 0	**PRI**: *SWIM access priority* This bit can be read or written through SWIM only. Usually the SWIM accesses to system resources are non-intrusive, SWIM having the lowest priority. This can be overridden by setting this bit. 0: Non-intrusive access by SWIM to system resources (low priority) 1: Intrusive access by SWIM to system resources (SWIM has priority, CPU is stalled). **Note:** The IOM bit is located in the STM8 core configuration register. Refer to the corresponding datasheet for information on this register

3.10.2 SWIM clock control register (CLK_SWIMCCR)

Address Offset: 50CDh (product dependent)

Reset value: xxxx 0000 (x0h)

7	6	5	4	3	2	1	0
Reserved							SWIMCLK
							rw

Bits 7:1	Reserved, must be kept cleared.
Bit 0	**SWIMCLK** *SWIM clock divider* This bit is set and cleared by software. 0: SWIM clock divided by 2 1: SWIM clock not divided by 2 **Note:** this register is not present in some STM8 devices.

4 Debug module (DM)

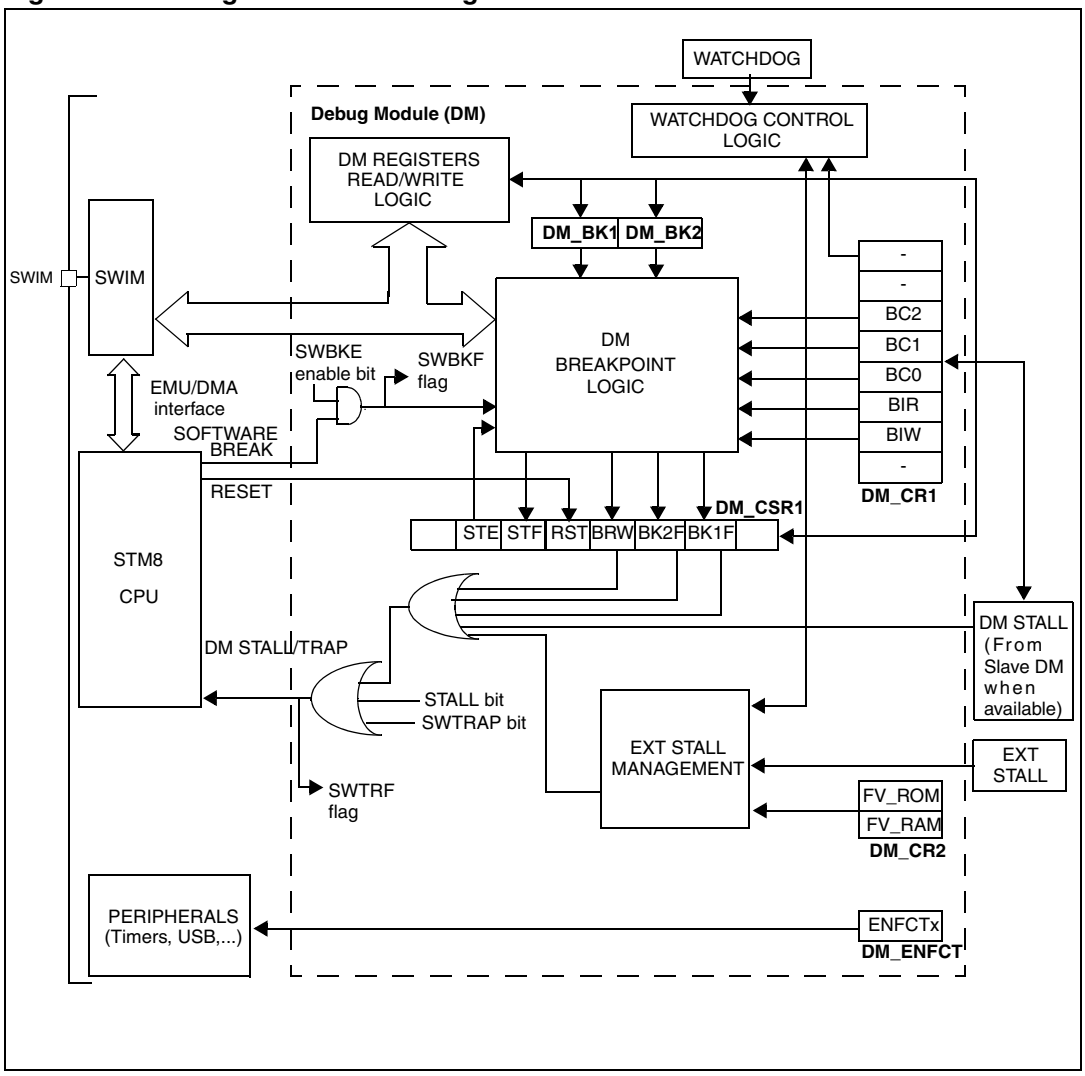
4.1 Introduction

The Debug Module (DM) allows the developer to perform certain debugging tasks without using an emulator. For example, the DM can interrupt the MCU to break infinite loops or output the core context (stack) at a given point. The DM is mainly used for in-circuit debugging.

4.2 Main features

- Two conditional breakpoints (break on instruction fetch, data read or write, stack access...)
- Software breakpoint control
- Step mode
- External Stall capability on WOTF command in SWIM mode
- Watchdog and peripherals control
- DM Version identification capability
- Interrupt Vector Table selection

Figure 11. Debug module block diagram



4.3 Debug

The DM registers can be read and written only through the SWIM interface. STM8 core has no access to these registers.

4.3.1 Reset

Once the SWIM is active and SWIM_DM bit is set in SWIM_CSR register, a ‘data read’ breakpoint at the reset vector address is automatically set, due to the reset values of the debug module registers. This breakpoint can be used to initialize the debug session.

4.3.2 Breakpoints

The DM generates a stall to the core when a breakpoint is reached. When the processor is stalled, the host can read or modify any address in memory. Access to the processor registers is explained in *Section 3.7: CPU register access*.

To restart the program execution, the STALL bit in DM_CSR2 must be cleared using the WOTF command of the SWIM protocol.

4.3.3 Abort

To use the Abort function, the host must write the STALL bit in the DM_CSR2 using the SWIM WOTF command.

No interrupt is generated. The core is stalled in the current state. Using the SWIM commands, the host can read and modify the status of the MCU. If the CPU registers must be modified, the procedure described in *Section 3.7: CPU register access* has to be used.

The host can restart the program execution by resetting the STALL bit using the SWIM commands.

4.3.4 Watchdog control

Using the WDGOFF bit in the *DM control register 1 (DM_CR1)* you can configure the Window Watchdog and Independent Watchdog counters to be stopped while the CPU is stalled by the Debug Module. This bit must be set before the watchdogs are activated. If a watchdog is enabled by Hardware Watchdog option bit, the WDGOFF bit has no effect on it.

4.3.5 Interaction with SWIM

The SWIM sends the status bit which indicates the SWIM is active or not. When SWIM is not active, the DM will not generate any break/stall request to the CPU.

4.4 Breakpoint decoding table

Table 4. Decoding table for breakpoint interrupt generation

DM_CR1					BREAK CONDITIONS	DM_CSR1		
BC2	BC1	BC0	BIR	BIW		BK1F	BK2F	BRW
0	0	0	0	0	Disabled (RESET state)	0	0	x
0	0	0	0	1	Data Write on @=BK1 and Data=BK2L	1	0	0
0	0	0	1	0	Data Read on @=BK1 and Data=BK2L	1	0	1
0	0	0	1	1	Data R/W on @=BK1 and Data=BK2L	1	0	0/1
0	0	1	0	0	Instruction fetch BK1<=@<=BK2	1	0	x
0	0	1	0	1	Data Write on BK1<=@<=BK2	1	0	0
0	0	1	1	0	Data Read on BK1<=@<=BK2	1	0	1
0	0	1	1	1	Data R/W on BK1<=@<=BK2	1	0	0/1
0	1	0	0	0	Instruction fetch on @<= BK1 or BK2<=@	1	0	x
0	1	0	0	1	Data Write on @<= BK1 or BK2<=@	1	0	0
0	1	0	1	0	Data Read on @<= BK1 or BK2<=@	1	0	1
0	1	0	1	1	Data R/W on @<= BK1 or BK2<=@	1	0	0/1
0	1	1	X	X	Disabled	0	0	x
1	0	0	0	0	Instruction fetch on @=BK1 then on @=BK2	0	1	x
1	0	0	0	1	Data Write on @=BK1 or @=BK2	10 or 01 or 11		0

Table 4. Decoding table for breakpoint interrupt generation

DM_CR1					BREAK CONDITIONS	DM_CSR1		
BC2	BC1	BC0	BIR	BIW		BK1F	BK2F	BRW
1	0	0	1	0	Data Read on @=BK1 or @=BK2	10 or 01 or 11		1
1	0	0	1	1	Data R/W on @=BK1 or @=BK2	10 or 01 or 11		0/1
1	0	1	0	0	Instruction fetch on @=BK1 or @=BK2	10 or 01 or 11		x
1	0	1	0	1	Instruction fetch on @=BK1 / Data Write on @=BK2	10 or 01		x-0
1	0	1	1	0	Instruction fetch on @=BK1 / Data Read on @=BK2	10 or 01		x-1
1	0	1	1	1	Instruction fetch on @=BK1 / Data R/W on @=BK2	10 or 01		x-0/1
1	1	0	X	X	Disabled	0	0	x
1	1	1	0	0	Data Write in Stack on @<=BK1 / Instruction fetch on @=BK2	10 or 01		0-x
1	1	1	0	1	Data Write in Stack on @<=BK1 / Data Write on @=BK2	10 or 01 or 11		0
1	1	1	1	0	Data Write in Stack on @<=BK1 / Data Read on @=BK2	10 or 01		0-1
1	1	1	1	1	Data Write in Stack on @<=BK1 / Data R/W on @=BK2	10 or 01 or 11		0-0/1

4.5 Software breakpoint mode

Software breakpoint mode is reserved for debugging tools to insert breakpoints into user code by substituting a user instruction with a software break (reserved BKPT instruction #8b).

Software breakpoint mode is enabled using the SWBKPE bit in the *DM control/status register 2 (DM_CSR2)*.

When a BKPT instruction is decoded, the CPU is stalled and the STALL and SWBKF bits are set by hardware to indicate that a software breakpoint has occurred. To resume execution, the debugger must restore the user's instruction, then set the FLUSH bit and clear the STALL bit.

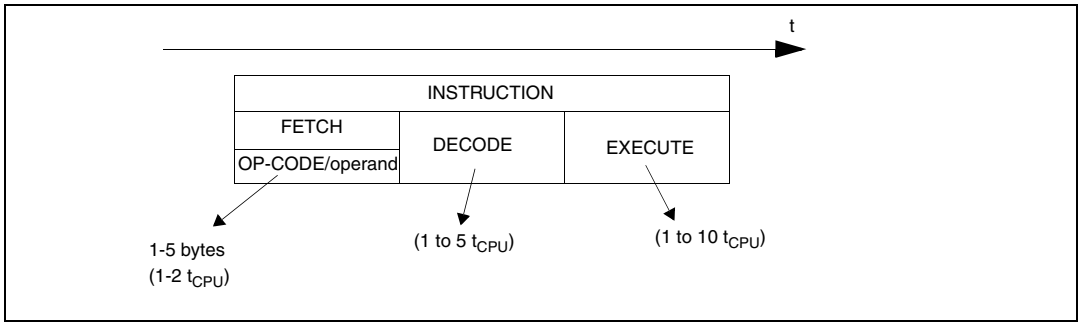
4.6 Timing description

This paragraph defines when the Debug Module stalls the CPU when using the different breakpoint sources.

- The STM8 instruction can be modelized in time with an op-code/operand FETCH phase DECODE and EXECUTION phases as shown in *Figure 12*.

The timing information is based on this models.

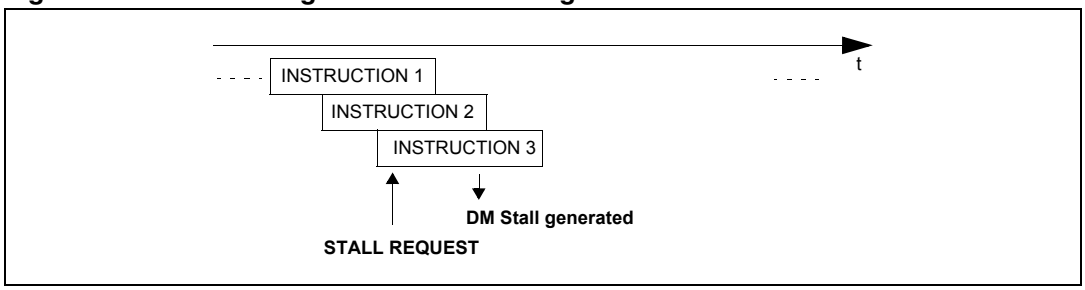
Figure 12. STM8 Instruction Model



4.7 Abort

The stall is generated immediately on writing the STALL bit in the DM_CSR2 register.

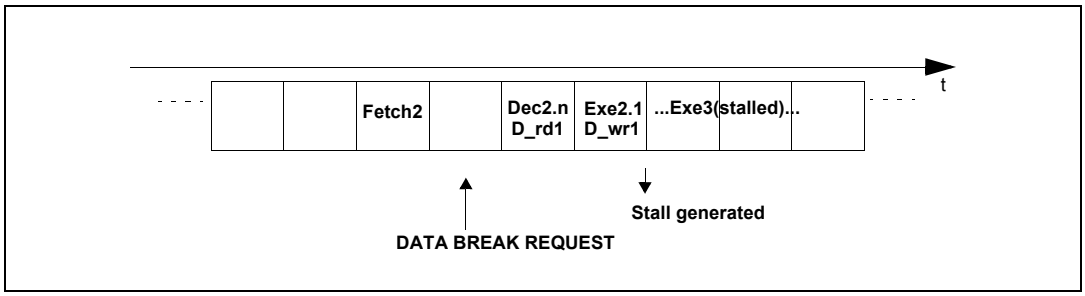
Figure 13. STM8 Debug Module Stall Timing



4.8 Data breakpoint

A stall is generated when SWIM is active, after the end of the current instruction execution.

Figure 14. STM8 DM Data Break Timing

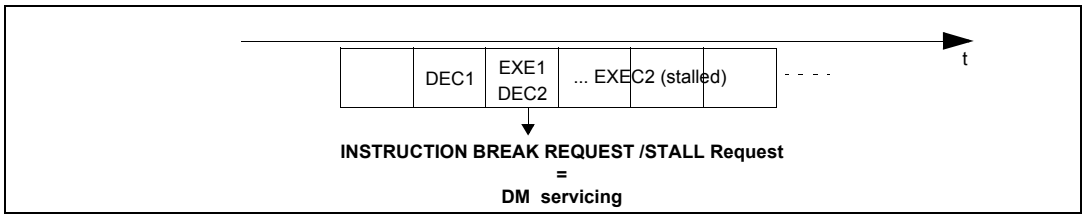


4.9 Instruction breakpoint

In the STM8, on an instruction break, DM stalls the CPU before the selected instruction execution (while the instruction is in the decode stage). See *Figure 15*.

Note: When the specified address does not correspond to a valid instruction address, no stall is generated.

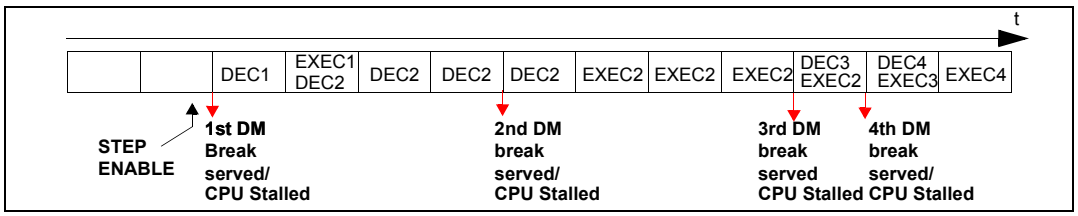
Figure 15. STM8 DM instruction break timing



4.10 Step mode

The STM8 CPU stall is activated before the instruction execution, in the first decode cycle of the instruction. See *Figure 16*.

Figure 16. STM8 DM step timing



Note: When Step mode and Instruction Break on the next instruction mode are both enabled, both the STF and the BKxF flags are set. When you clear the STALL bit, the step function continues its normal operation.

4.11 Application notes

4.11.1 Illegal Memory access

To verify if the program attempts to write or read in an illegal part of memory (reserved area), select the "Data R/W on BK1<=@<=BK2" condition, where BK1 and BK2 are the lower and upper addresses of the reserved memory.

4.11.2 Forbidden stack access

If part of the stack contains specific data or instructions that should not be overwritten, the DM can be used to prevent access to these locations. Select one of the "Data Write in Stack on @<=BK1" conditions and set BK1 to the upper value where the specific data are located in the stack. If the STM8 tries to overwrite these values (after an interrupt or a CALL...), DM will generate a break. The four possible associated conditions allow to manage another breakpoint capability at the same time.

4.11.3 DM break

After an DM break, the CPU is stalled (through the EMU_Stall signal). While the CPU is stalled, the SWIM can read/write any memory location or memory mapped register.

The program can be continued from the breakpoint, by resetting the Stall bit.

If a change of PC is needed, the SWIM must write the new PC value using the method described in *Section 3.7: CPU register access*. In order to fetch the code from the new PC address, the SWIM must set the FLUSH bit in the *DM control/status register 2 (DM_CSR2)* before resetting the STALL bit .

4.12 DM registers

These registers are read/write only through the SWIM interface.

In this section, the following abbreviations are used:

read/write (rw)	SWIM can read and write to these bits via the ROTF/WOTF commands.
read-only (r)	SWIM can only read these bits via the ROTF command.

4.12.1 DM breakpoint register 1 extended byte (DM_BKR1E)

STM8 Address: 7F90h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK1[23:16]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK1[23:16]**: *Breakpoint 1 extended byte value* This register is written by software to define the extended 8 address bits of Breakpoint 1.

4.12.2 DM breakpoint register 1 high byte (DM_BKR1H)

Address: 7F91h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK1[15:8]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK1[15:8]**: *Breakpoint 1 high byte value* This register is written by software to define the higher 8 address bits of Breakpoint 1.

4.12.3 DM breakpoint register 1 low byte (DM_BKR1L)

Address: 7F92h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK1[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK1[7:0]**: *Breakpoint 1 high byte value* This register is written by software to define the lower 8 address bits of Breakpoint 1.

4.12.4 DM breakpoint register 2 extended byte (DM_BKR2E)

Address: 7F93h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK2[23:16]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK2[23:16]**: *Breakpoint 2 extended byte value* This register is written by software to define the extended 8 address bits of Breakpoint 2.

4.12.5 DM breakpoint register 2 high byte (DM_BKR2H)

Address: 7F94h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK2[15:8]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK2[15:8]**: *Breakpoint 2 high byte value* This register is written by software to define the higher 8 address bits of Breakpoint 2.

4.12.6 DM breakpoint register 2 low byte (DM_BKR2L)

Address: 7F95h

Reset value: 1111 1111 (FFh)

7	6	5	4	3	2	1	0
BK2[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**BK2[7:0]**: *Breakpoint 2 high byte value* This register is written by software to define the lower 8 address bits of Breakpoint 2.

4.12.7 DM control register 1 (DM_CR1)

Address: 7F96h

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
WDGOFF	Reserved	BC[2:0]			BIR	BIW	Reserved
rw	-	rw	rw	rw	rw	rw	

Bit 7	**WDGOFF** *Watchdog control enable.* This bit must be set or cleared by software before the watchdogs (WWDG and/or IWDG) are activated. This bit has no effect if the hardware watchdog option is selected. 0: Watchdog counters are not stopped while CPU is stalled by DM 1: Watchdog counters are stopped while CPU is stalled by DM
Bit 6	Reserved.
Bits 5:3	**BC[2:0]** *Breakpoint control* These bits are set and cleared by software, they are used to configure the breakpoints as shown in *Table 4*.
Bit 2	**BIR** *Break on read control* This bit enables a breakpoint on a data read operation. It is set and cleared by software. 0: No break on data read 1: Break on data read
Bit 1	**BIW** *Break on write control* This bit enables a breakpoint on a data write operation. It is set and cleared by software. 0: No break on data write 1: Break on data write
Bit 0	Reserved**.**

4.12.8 DM control register 2 (DM_CR2)

Address: 7F97h

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved					FV_ROM	Reserved	FV_RAM
					rw		rw

Bit 7:3	These bits are reserved and must be kept at 0.
Bit 2	**FV_ROM** *Remap Vector table in ROM*. This bit is set or cleared by software. It remaps the vector table to a ROM location (product dependent) instead of program memory (usually 8000h). 0: Vector table is in Program Memory area (8000h) 1: Vector table is in ROM memory area (depends on the product)
Bit 1	Reserved, must be kept at 0.
Bit 0	**FV_RAM** *Remap vector table in RAM* This bit is set or cleared by software. It remaps the interrupt vector table to a RAM location instead of program memory (usually 8000h). 0: Vector table is in Program Memory area (8000h) 1: Vector table is in RAM memory area (address depends on the product)

4.12.9 DM control/status register 1 (DM_CSR1)

Address: 7F98h

Reset value: 0001 0000 (10h)

7	6	5	4	3	2	1	0
Reserved	STE	STF	RST	BRW	BK2F	BK1F	Reserved
	rw	rw	r	r	r	r	

Bit 7	Reserved.
Bit 6	**STE** *Step mode enable* (Read / Write) This bit is set and cleared by software. It enables Step mode. 0: Step mode disabled 1: Step mode enabled
Bit 5	**STF** *Step Flag* (Read Only) This bit indicates that the stall was generated by Step mode. It is set and cleared by hardware. Writing to this bit does not change the bit value. 0: Step mode stall did not occur 1: Step mode stall occurred
Bit 4	**RST** *Reset Flag* (Read Only) This bit is set by hardware when the CPU was stalled by the debug module (DM), just after reset. It is cleared by hardware when the STALL bit is cleared. Writing to this bit does not change the bit value. 0: No reset occurred 1: A reset occurred
Bit 3	**BRW** *Break on Read/Write Flag* (Read Only). This bit gives the value of the read/write signal when a break occurs. Its value is not significant for instruction fetch breaks. It is set by hardware depending on the breakpoint conditions (see *Table 4: Decoding table for breakpoint interrupt generation on page 20*) and is cleared by hardware depending on the next breakpoint conditions. Writing to this bit does not change the bit value. 0: Breakpoint on write 1: Breakpoint on read
Bit 2	**BK2F** *Breakpoint 2 Flag* (Read Only). This bit indicates that the DM stall was generated by Breakpoint 2. It is set by hardware depending on the control conditions (see *Table 4: Decoding table for breakpoint interrupt generation on page 20*) and is cleared by hardware when the STALL bit is cleared. Writing to this bit does not change the bit value. 0: Breakpoint 2 did not occur 1: Breakpoint 2 occurred
Bit 1	**BK1F** *Breakpoint 1 Flag* (Read Only). This bit indicates that the DM interrupt was generated by Breakpoint 1. It is set by hardware depending on the control conditions (see *Table 4: Decoding table for breakpoint interrupt generation on page 20*) and is cleared by hardware when the STALL bit is cleared. Writing to this bit does not change the bit value. 0: Breakpoint 1 did not occur 1: Breakpoint 1 occurred
Bit 0	Reserved

4.12.10 DM control/status register 2 (DM_CSR2)

Address: 7F99h

Reset value: 0000 0000 (00h)

7	6	5	4	3	2	1	0
Reserved		SWBRK	SWBKF	STALL	Reserved		FLUSH
		rw	r	r			rw

Bits 7:6	Reserved. Must be kept at 0
Bit 5	**SWBKE** *Software breakpoint control bit* (read/write) This bit is used to enable/disable the software breakpoint capability with NOP instruction 0: DM does not generate any event when NOP(SW BRK) instruction is fetched by CPU 1: DM generates an event (CPU stalled in SWIM mode) when a software break instruction is fetched by CPU.
Bit 4	**SWBKF** *Software breakpoint status bit* (read only) This flag is set when the CPU executes the software break instruction. 0: No software break instruction detected. 1: Software break instruction detected. This bit is cleared when the STALL bit is cleared.
Bit 3	**STALL** *CPU stall control bit* (R/W only in SWIM mode) This bit is used to stall the CPU. This bit is kept cleared if the device is not in SWIM mode. This bit is set by WOTF command to generate an ABORT equivalent command It is also set by an DM trap interrupt event. This bit is cleared by WOTF command to re-start the CPU. 0: CPU runs normally 1: CPU is stalled
Bit: 2:1	Reserved. Must be kept at 0
Bit: 10	**FLUSH** *Flush decode* This bit is set by software to flush the instruction decode phase after a PC modification. It is cleared by hardware when the flush is completed. 0: Default status 1: Flush decode

4.12.11 DM enable function register (DM_ENFCTR)

Address: 7F9Ah

Reset Value: 1111 1111 (FFh)

7							0
ENFCT7	ENFCT6	ENFCT5	ENFCT4	ENFCT3	ENFCT2	ENFCT1	ENFCT0
rw	rw	rw	rw	rw	rw	rw	rw

Bits 7:0	**ENFCTx** *Enable function* This bit is set and cleared by software. it allows to freeze a particular function of a peripheral when the core is stalled. The ENFCTx bit definitions are product dependent. 0: Function is frozen when CPU is stalled by DM 1: Function is active

4.12.12 Summary of SWIM, DM and core register maps

Table 5. STM8 registers

STM8 Address	Register Name	7	6	5	4	3	2	1	0
7F00h	A Reset value	A7 0	A6 0	A5 0	A4 0	A3 0	A2 0	A1 0	A0 0
7F01h	PCE(1)	PC23	PC22	PC21	PC20	PC19	PC18	PC17	PC16
7F02h	PCH(1)	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8
7F03h	PCL(1)	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
7F04h	XH Reset value	X15 0	X14 0	X13 0	X12 0	X11 0	X10 0	X9 0	X8 0
7F05h	XL Reset value	X7 0	X6 0	X5 0	X4 0	X3 0	X2 0	X1 0	X0 0
7F06h	YH Reset value	Y15 0	Y14 0	Y13 0	Y12 0	Y11 0	Y10 0	Y9 0	Y8 0
7F07h	YL Reset value	Y7 0	Y6 0	Y5 0	Y4 0	Y3 0	Y2 0	Y1 0	Y0 0
7F08h	SPH(1)	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8
7F09h	SPL(1)	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
7F0Ah	CC Reset value	V 0	- 0	I1 1	H 0	I0 1	N 0	Z 0	C 0
7F80h	SWIM_CSR Reset value	SAFE_MASK 0	NO_ACCESS 0	SWIM_DM 0	HS 0	OSCOFF 0	RST 0	HSIT 0	PRI 0
7F90h	DM_BK1RE Reset value	BK1R23 1	BK1R22 1	BK1R21 1	BK1R20 1	BK1R19 1	BK1R18 1	BK1R17 1	BK1R16 1
7F91h	DM_BK1RH Reset value	BK1R15 1	BK1R14 1	BK1R13 1	BK1R12 1	BK1R11 1	BK1R10 1	BK1R9 1	BK1R8 1
7F92h	DM_BK1RL Reset value	BK1R7 1	BK1R6 1	BK1R5 1	BK1R4 1	BK1R3 1	BK1R2 1	BK1R1 1	BK1R0 1
7F93h	DM_BK2RE Reset value	BK2R23 1	BK2R22 1	BK2R21 1	BK2R20 1	BK2R19 1	BK2R18 1	BK2R17 1	BK2R16 1
7F94h	DM_BK2RH Reset value	BK2R15 1	BK2R14 1	BK2R13 1	BK2R12 1	BK2R11 1	BK2R10 1	BK2R9 1	BK2R8 1
7F95h	DM_BK2RL Reset value	BK2R7 1	BK2R6 1	BK2R5 1	BK2R4 1	BK2R3 1	BK2R2 1	BK2R1 1	BK2R0 1
7F96h	DM_CR1 Reset value	WDGOFF 0	Reserved 0	BC2 0	BC1 0	BC0 0	BIR 0	BIW 0	Reserved 0
7F97h	DM_CR2 Reset value	Reserved					FV_ROM 0	Reserved 0	FV_RAM 0
7F98h	DM_CSR1 Reset value	Reserved 0	STE 0	STF 0	RST 0	BRW 0	BK2F 0	BK1F 0	Reserved 0
7F99h	DM_CSR2 Reset value	Reserved 0	Reserved 0	SWBKE 0	SWBKF 0	STALL 0	Res	Reserved 0	FLUSH 0
7F9Ah	DM_ENFCTR Reset value	ENFCT7 1	ENFCT6 1	ENFCT5 1	ENFCT4 1	ENFCT3 1	ENFCT2 1	ENFCT1 1	ENFCT0 1

1. The reset value for the SP and PC registers is product dependent. Refer to the device datasheet for more details

5 Revision history

Table 6. Document revision history

Date	Revision	Changes
15-Jan-2008	1	Initial release.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com