ST7 FAMILY

ST72Exx JTAG PROGRAMMING SPECIFICATIONS

This document contains all the information needed to program any device in the ST7 general purpose microcontroller family.

The programming can be done in two different ways:

1) JTAG mode: by loading a code in the ST7 internal RAM which will be executed by the ST7 to program itself. The loading of the program in RAM can be done using the JTAG interface. This method has the advantage of being very general. Moreover it requires just a few pins and is therefore mandatory for ST7 devices with a very low pincount.

2) Bootstrap mode: by forcing the ST7 to execute an external code that will program its memory. This method will make use of the bootstrap loader feature which requires a more extended interface (address & data buses). It can be used only for those members of the family where these signals are available at pin level. This method is normally be slightly faster.

This document first describes the ST7 Serial Test Controller (STC) and test modes, then provides guidelines on how to load a program in internal RAM via the JTAG interface and how to program the EPROM memory. Examples and cross reference tables are given at the end of the document.

For further details about device characteristics, please refer to the programming manual and the data book for ST7 family.

Rev. 1.0

February 1998

Table of Contents

Table of Contents

1 INTRODUCTION

1.1 DOCUMENT CONVENTIONS

In the equations given in this document, in these equations the following conventions are used:

- The symbol ”*” (star) means logical AND
- The symbol ”+” (plus) means logical OR
- The symbol ”/” (slash) means logical NOT

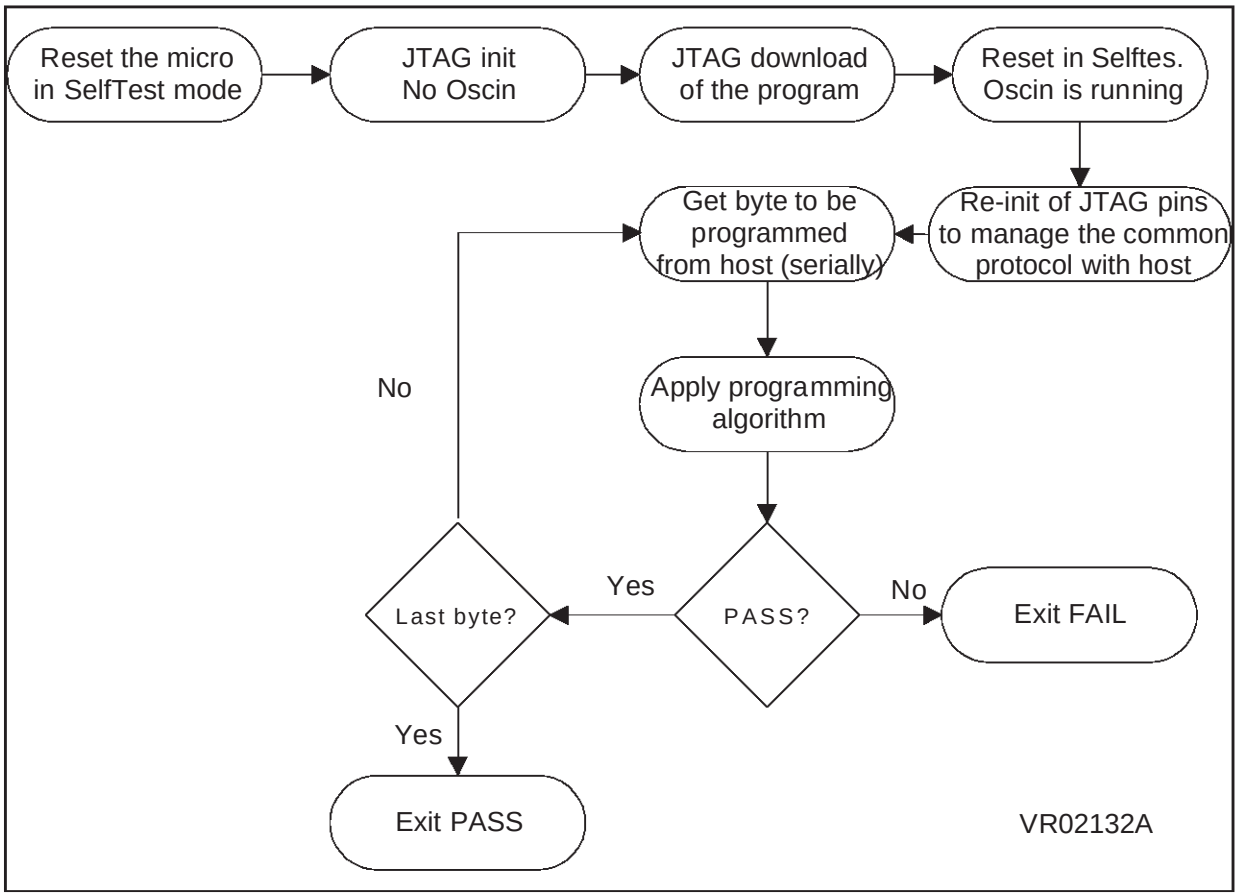
1.2 JTAG MODE PROGRAMMING FLOW OVERVIEW

To program the ST7 EPROM via the JTAG, you must control at least the following 11 pins:

- System pins: OSCIN, RESETN
- JTAG interface (5 pins)
- 4 pins to force the test modes: M0, M1, M2 and V_{PP}/TEST

Below you will find a flow chart of operations; when reading it, keep in mind that:

- all operations from the 5th step on are performed by the program that was loaded with steps one to four.
- reset means hardware reset, that is pulling low the reset pin of the device.



1.3 BOOTSTRAP MODE PROGRAMMING FLOW OVERVIEW

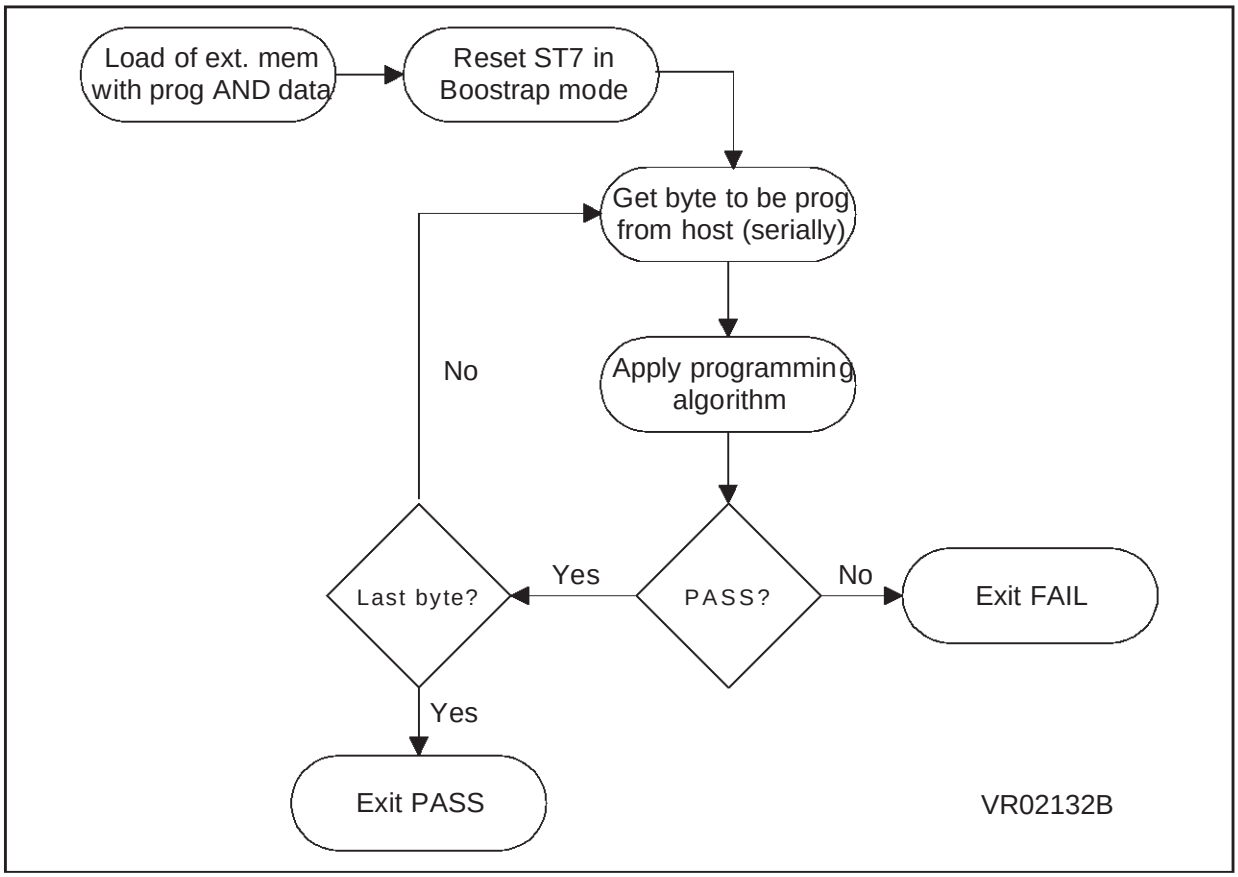
To program the ST7 EPROM via the bootstrap mode, you must control at least the following 16 pins (considering an 8 bit address):

- System pins: RESETN
- Address and data buses
- Control signals (PH1, NESEL, RW)
- 4 pins to force the test modes: M0, M1, M2 and V_{PP}/TEST

If any of these signals are not available on a pin this method can not be used.

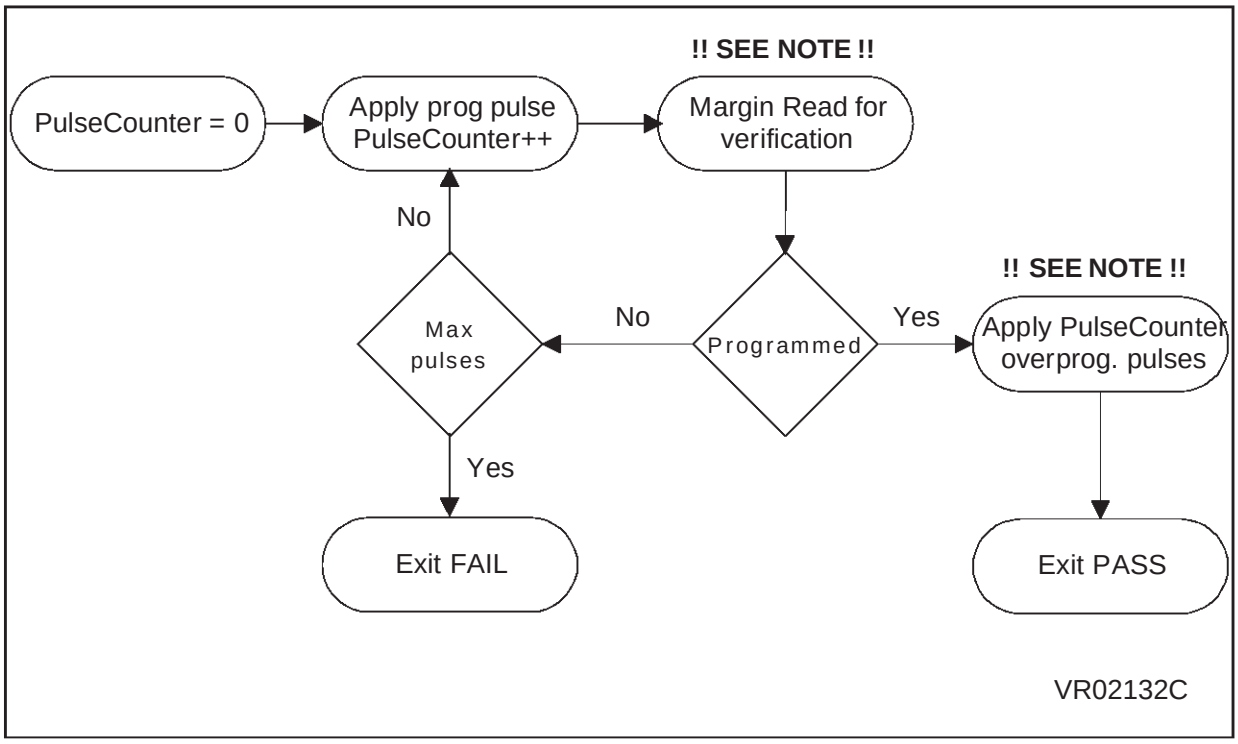
Here follows a flow chart of operations

:



1.4 EPROM PROGRAMMING ALGORITHM OVERVIEW

To optimize the programming time and to reduce the stress on the EPROM, use the following programming algorithm:



Note: The overprogramming pulses must be applied only if Margin Read is not available (on some old devices) because a standard read is not safe enough in this case.

2 SERIAL TEST CONTROLLER DESCRIPTION

2.1 ABSTRACT

The Serial Test Controller (STC) macrocell allows an integrated access to internal test and boundary scan test features, based on Joint Test Action Group (JTAG) standards (IEEE 1149.1).

It operates independently from the remaining part of the device, loading test instructions and data through a serial Test Access Port (TAP), which allows also test results to be read out. Test instruction execution is controlled through signals applied to Test Mode Select (TMS) and Test Clock (TCK) input pins.

The STC supports the Boundary Scan test technique. Therefore, all the pins of the ST7 are connected to a special shift register controlled by the STC. This register allows signals at component boundary to be forced or observed using scan testing principles.

2.2 NOTICE

This document is not intended to explain operation principles of the circuit based on the IEEE 1149.1 standard, to which reference must be made for details.

2.3 ARCHITECTURE

The STC contains 3 main functional blocks:

- Test Access Port (TAP):
  Five dedicated pins: TDI, TDO, TRST0, TCK and TMS are used to read/write Scan or Instruction registers and supply sequences that control the test logic.
- Instruction Block:
  Built around a shift-based 4-bit Instruction Register and an Instruction Decoder. It is serially loaded with patterns, and selects the test operation to be performed.
- Scan Registers:
  Test data or stimuli required by a test are serially loaded into the register selected by the current instruction, and can be shifted out after completion of the test operation

The STC is running in the following condition:
V_{PP}_TEST * (JTAG_EN + ROUTRAM + SELFTEST + ROMDUMP + OPEN + CPUTEST)

2.4 PINS

The STC is designed so that undriven inputs, caused by tristate drivers or open-circuit faults, produce a deterministic behaviour. This is to ensure deterministic STC operation in any condition, and is implemented by weak pull-up resistors on input buffers.

Five pins allow access to serial test control features:

- **TCK**: Test Clock (input)
Clock of the STC, independent from System clock. Test logic retains its previous state when the TCK signal is stopped in a low or high state.

- **TMS**: Test Mode Select (input, weak pull-up)
Serial Input, sampled on the rising edge of TCK. The TMS signal selects a Scan Register or the Instruction Register to be shifted in/out through TDI/TDO. The behaviour of the STC is described by the state-diagram given below (Figure 1). The TMS signal is held at “1” if undriven, so that the STC remains in the Test-Reset status, and doesn’t interfere with normal circuit operation.

- **TDI**: Test Data In (input, weak pull-up)
Input pin of the serial data line, sampled on the rising edge of TCK. The TDI signal provides data to be shift in the Instruction Register or an other scan chain. TDI is held at “1” if undriven, so that the BYPASS instruction (opcode = Fh) is selected, and the normal circuit operation not affected.

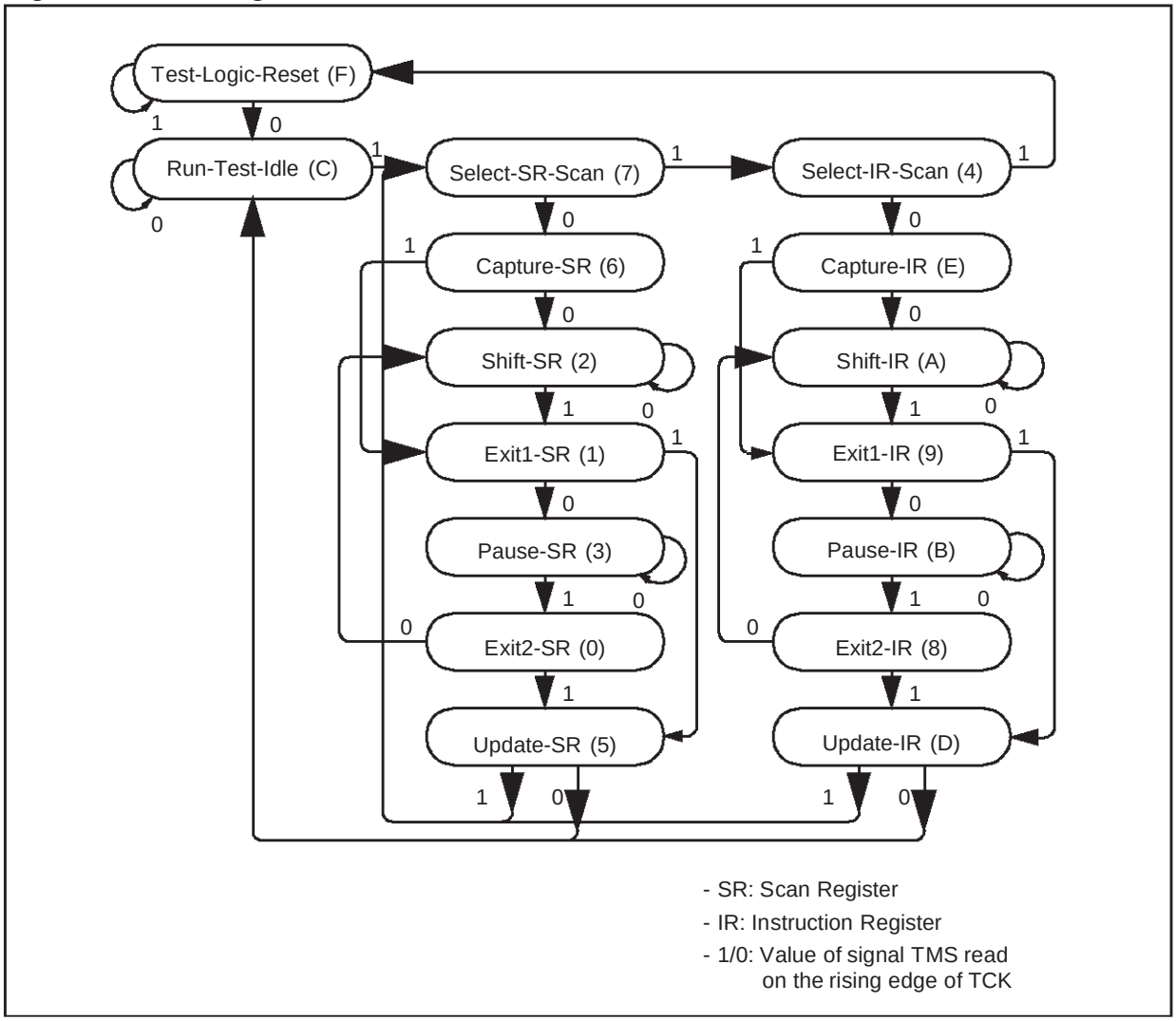
- **TDO**: Test Data Out (tristate output)
Output of the serial data line, updated on the falling edge of TCK. Data appears with no inversion with respect to TDI. The signal is tristate, and is held inactive except when scanning of test registers is in progress. This is necessary since TDO signals coming from different chips on a board may be connected in parallel.

- **TRST0**: Reset (Input, weak pull-up, active low)
Asynchronous and immediate Reset input of the STC. This pin should be held at “0” during power-up by users to disable all test logic and enter normal mode.

This pin is held at “1” if undriven, allowing STC operation to take place under control of the TCK and TMS inputs. This pin has no effect on device logic other than STC.

If the JTAG function is requested by the customer, these 5 pins must be dedicated to this function.

Figure 1. State diagram of the STC



2.5 OPERATION

The STC controller is a 16-state finite state machine that operates according to the state-diagram given above (Figure 1). Transitions from one state to another take place on the rising edges of the TCK clock according to the value of the TMS signal. The TMS signal must be updated by the user after the falling edge of TCK.

The behaviour of the controller in each state is completely defined in the IEEE 1149.1 standard.

2.6 INSTRUCTIONS

The primary effect of JTAG instructions is to select one of the scan chains, and to connect it between the TDI and TDO pins. This allows you to serially read/write data into any scan register.

Instructions that are implemented in the STC follow both IEEE rules and ST7 specifications. Some instructions are required by IEEE standards as mandatory for a compliant device, others are indicated as optional. All mandatory instructions are implemented, along with some optional ones.

Opcode	Instruction	Selected Register	Number of TCK cycles	Type
1111 (Fh)	BYPASS	BYPASSR	17	
0000 (0h)	EXTEST	BSR	16 + L (BSR)	
0011 (3h)	SAMPLE/RELOAD	BSR	16 + L (BSR)	
0001 (1h)	IDCODE	IDCODR	48	
1100 (Ch)	CLAMP	BYPASSR	17	Optional
1101 (Dh)	HIGHZ	BYPASSR	17	
Others	Reserved	BYPASSR	-	Test Use only
L(SC) = number of stages in the scan chain SC				

IMPLEMENTED INSTRUCTIONS

IEEE Mandatory Instructions:

- BYPASS: The TDI and TDO pins are connected by a one-stage shift register. This establishes a minimum-length serial path between TDI and TDO, to bypass the chip in a multichip-on-board test situation.

- SAMPLE/PRELOAD: Used to transparently observe/preset the signal values on device pins. The instruction connects the Boundary Scan Register (BSR) between TDI and TDO, and is performed in 3 steps:

- SAMPLE phase: Pin values are latched in the BSR.

- SHIFT phase: Serial scan of the BSR takes place transparently with respect to normal chip operation, allowing to read the strobed values, and loading at the same time the BSR with a new set of values.

- PRELOAD phase: as soon as data shifting is completed, the new values are latched on the Parallel Output of the BSR cells. These values are ready to be forced on the pins of the device if for instance a CLAMP instruction is performed.

SAMPLE/PRELOAD instruction execution is transparent with respect to device operation, since strobing, shifting and updating of the BSR do not influence any system signal or pin.

- EXTEST: One-shot test of off-chip connections and circuitry. The operation follows the same flow as the SAMPLE/PRELOAD instruction: SAMPLE, SHIFT, PRELOAD. The difference with the SAMPLE/PRELOAD instruction is that during execution of the EXTEST instruction, pin values are controlled by the test logic. Values on output pins of the chip, and input to the device logic, are the ones previously loaded into the BSR. This allows testing of board-level interconnections, and scanning for possible damaged connections in a multichip test situation. Usually one or a sequence of EXTEST instructions, with different test vectors to test board interconnections, is preceded by a SAMPLE/PRELOAD instruction, to load the first vector transparently. This vector is forced onto output pins of the device as soon as the EXTEST execution starts. BSR values are forced onto pins on the falling edge of TCK in the Update-IR controller state after loading of an EXTEST instruction. The pins are released in the same state, after loading a different instruction in the STC Instruction Register.

IEEE Optional Instructions:

- IDCODE: Connects the IDCODE Shift Register between TDI and TDO, to shift out the 32-bit device Identification Code, containing manufacturer, product, version data of the device.

- CLAMP: Device output pins are forced with the values contained in the BSR. These values are the ones loaded with a previous SAMPLE/PRELOAD or EXTEST instruction. The BYPASS register is connected between pins TDI and TDO.

- HIGHZ: Device bidirectional and output tristate pins are held in the high-impedance state. The BYPASS register is connected between pins TDI and TDO.

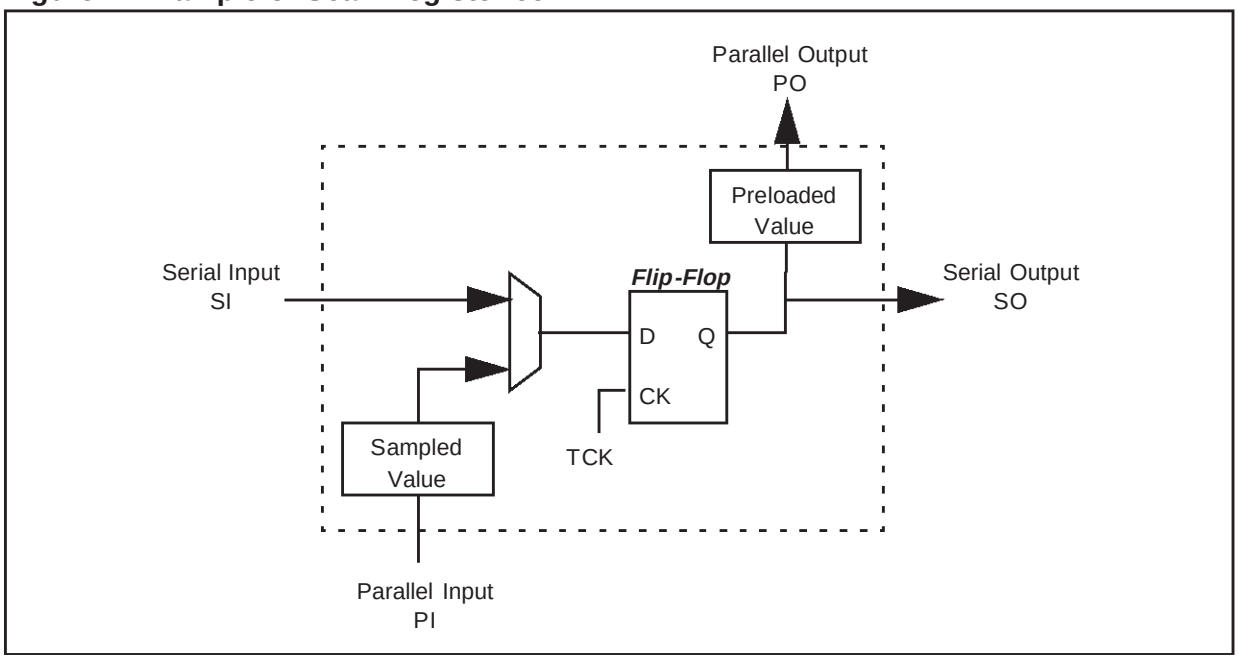
2.7 SCAN REGISTERS

The following IEEE 1149.1 standard scan registers are implemented as shift registers, inside the Serial Test Controller (STC), accessible through the Test Access Port (TAP). Because of this, they are not accessible by standard software addressing, unlike any other ST7 registers. Only the CRC is accessible by software.

2.8 GENERAL REGISTER ARCHITECTURE

Each scan register is composed of a chain of cells. Shift operation is enabled through the Serial Input (SI) and Output (SO) when the STC is in its Shift-SR state. Serial Output of the last element is output on TDO on the falling edge of TCK. TDI value is loaded on the subsequent rising edge of TCK.

Figure 2. Example of Scan Register cell



Scan Register cells may also have a Parallel Input (PI) and Output (PO), allowing the loading of every cell in parallel, or to force the contents of the register cells into the system logic connected to the register. See Figure 2.

The Parallel Input, when present, initialises the register before shifting. It is activated on the TCK low phase, in the Capture-SR state.

The Parallel Output may be latched, allowing to output a constant value during register shifting and to synchronise the moment in which register contents are forced into the system logic. If not specified otherwise, updating of latched PO is performed on the TCK low phase in the Update-SR state, once shifting of the register is finished.

If not specified otherwise, there is no inversion between data input and data output of scan registers cells, and thus between TDI and TDO.

Each register has a fixed length.

2.9 INSTRUCTION REGISTER

The instruction register contains the opcode of the JTAG instruction currently performed by the STC. Opcodes are entered serially through TDI in the select-IR-scan state of the STC.

Here are the main features of the Instruction Register:

- 4-bit register (up to 16 instructions).

- Latched PO, to the STC Instruction Decoder.

- All actions resulting from an instruction (for example EXTEST) finish when a different instruction is latched in the PO.

- The IDCODE instruction is automatically forced in the Instruction Register after a Test Logic Reset. This ensures that consistent data is always present on TDO, and that test logic does not interfere with the normal system operation. In addition blind reading of the Device Identification Code can be performed by the customer, with undriven TDI and TMS pins.

- Opcode of instruction IDCODE “0001” is also forced at the beginning of each instruction fetch cycle. This complies with rule 6.1.1_d, that requires the loading of bits 1 and 0 (LSB) with “01”. These bits are the first 2 to be shifted out during Instruction Code loading, allowing a simple check of fault isolation on the board level serial data path.

- Opcode of instruction BYPASS is forced on the Output of the Instruction Register Decoder when an unused opcode is entered to ensure deterministic operation of STC.

2.10 BYPASS REGISTER

The BYPR is a one-stage shift register, used to bypass the device in a multichip serial test environment. It’s contents is pre-set to “0” before any bypass scan operation.

2.11 CONFIDENTIAL TESTING FEATURES

2.11.1 Confidential instructions

Confidential instructions are used to access test features, and are neither documented nor accessible to customers.

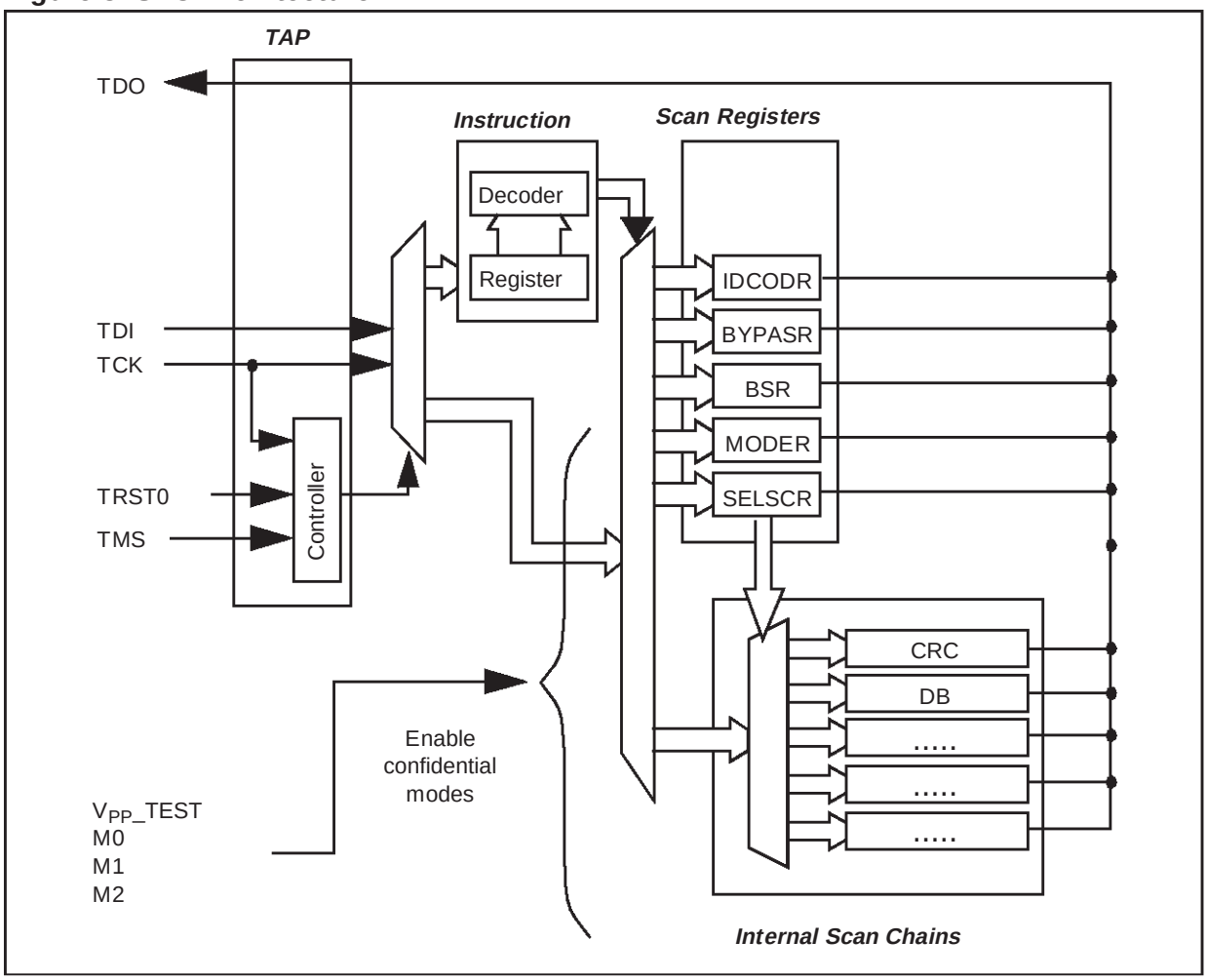
Opcode	Instruction	Selected Register	Number of TCK cycles	Type
1111 (Fh)	BYPASS	BYPASSR	17	
0000 (0h)	EXTEST	BSR	16 + L (BSR)	
0011 (3h)	SAMPLE/RELOAD	BSR	16 + L (BSR)	
0001 (1h)	IDCODE	IDCODR	48	
1100 (Ch)	CLAMP	BYPASSR	17	Optional
1101 (Dh)	HIGHZ	BYPASSR	17	
0110 (6h)	MODE	MODER	28	
1000 (8h)	SELSCAN	SELSCR	20	Confidential
0100 (4h)	INTSCAN	Internal chain	16 + L(chain)	
others	Reserved for future expansion			

- MODE: Selects the MODER JTAG register, composed of flags controlling various test configurations, and connects it between TDI and TDO for shifting.

- SELSCAN: Pre-selects the internal scan chain to be accessed. Due to the number of scan chains present, an indirect selection is needed: the SELSCR JTAG register is selected by this instruction and may be loaded with the code of the internal chain requested. That chain is then made active by the next INTSCAN JTAG instruction.

- INTSCAN: Enables the internal scan chain currently selected by SELSCR JTAG register contents, and connects it between TDI and TDO for shifting.

Figure 3. STC Architecture



2.11.2 Scan registers

Scan Registers act as control registers for test configuration and mode selection.

SCAN CHAIN SELECTION REGISTER

SELSCR is a 4-bit register enabled by the SELSCAN JTAG instruction. SELSCR is used to select which of the internal scan chains will be enabled by the next INTSCAN JTAG instruction.

Reset value: 0000, no chain is selected.

Selscr	Chain Selected	Length	Note
0000 (0h)	none	-	Reset Status
1010 (Ah)	DB	8	
1000 (8h)	CRC	16	
others	Reserved for peripheral scan chains		

MODE REGISTER (MODER)

MODER is an 12-bit register enabled by the MODE JTAG instruction. MODER contains Configuration Mode Flags that control various configuration and test mode features.

Reset Value: 0000 0000 0000, all flags are inactive

11											0
GPTC4	GPTC3	GPTC2	GPTC1	GPTC0	CRCSER	CPUBKPG	TSTBKP	NOMUX	FASTRAN	LOADDB	LOADIO

Bit [11:7] = **GPTC[11:7]** *General Purpose Test Control*
These 5 bits can be dedicated to a particular test feature in a product.

Bit11 = CKDIV2: If set, the internal clock PHI1 is equal at OSCIN divided by 2.

Bit6 = **CRCSER** *CRC in SERial mode*
This bit is active only when a scan chain, except the CRC, is selected (SELSC not equal to 0000 nor to 1000). When set, the CRC analyses TDO instead of DB and address bus. This is used to reduce the pattern length of macrocells which are tested with a full scan approach.

Bit5 = **CPUBKP** *test of CPU Bus KeePer When set*
The test of the bus keeper inside the CPU can be performed. The execution of the instructions is corrupted during this test.

Bit4 = **TSTBKP** *TeST Bus KeePer*
If set, a read command of a register is inhibited in each macrocell. A read of these registers allow to test the bus keeper on the internal data bus of each macrocell.

Bit3 = **NOMUX** *address/data NOt MUltipleXed in open modes*
In OPEN, BOOTSTRAP, ROMDUMP and CPUTEST modes, address and data are normally multiplexed on an I/O port for external accesses. If NOMUX is set, the data and address are not multiplexed, only the data bus (DB) is present.

Bit2 = **FASTRAN** *FAST TRANsfer*
If set, the instruction register of CPU is forced a specific opcode for fast memory dump/ load. In this mode the CPU does fetch a new opcode from the data bus at the end of the current instruction. The address is decremented on each cycle. When FASTRAN is set, the value of RW signal depends on LOADDB and LOADIO bits:

- If LOADDB or LOADIO is set, RW signal is low for a memory load operation. The core does not drive the data bus in this case.

- If LOADDB and LOADIO are reset, RW signal is high for a memory dump operation.

Bit1 = **LOADDB** *LOAD data from DB scan register*
If set, the data are provided by the DB scan register

Bit0 = **LOADIO** *LOAD data from an I/O port*
If set, the data are provided in parallel by an I/O port.

2.11.3 Internal scan chains

Internal scan chains of the ST7 are addressed indirectly through the SELSCR register with the INTSCAN JTAG instruction. Therefore, before an internal scan chain read/write operation the SELSCR has to be previously loaded via a SELSCAN JTAG instruction.

CRC Scan Chain

The Cyclic Redundancy Check (CRC) is a16-bit Multiple Input Scan Register. In test modes the CRC analyses TDO or the data bus and the 8 least significant bits of the address bus according to the CRCSER-bit. It can be read at any time by the CPU. It can be reset to #0000 by writing in the least significant byte. The behaviour of the CRC depends on the operating mode; it is described in detail in OPERATING MODES chapter.

DB Scan Chain

The DB scan chain is a 8-bit Multiple Input Scan Register. It allows to perform serial load of the memory or to provide data to the CPU by driving the internal data bus (DB). The DB scan chain drives the data bus on the following condition:
LOADDB * FASTRAN * RW/ * PHI1 serial memory load
+ LOADDB * FASTRAN /* NDBRSEL / * RW * PHI1 serial data/instruct. for CPU.
The memories are disabled on LOADDB * FASTRAN/ using MEMOFF signal.

2.11.4 Memory fast dump, load

Fast dump/load of the internal memory (RAM, ROM, registers) can be achieved by managing the STC. For this, the STC provides means to control the CPU and to use the I/O port dedicated to output address / data in open mode, to load in / dump out data.

Fast Dump/load procedures need the data memory start address. This value can be pre-set executing normal CPU instructions (coming from software or from the STC interface). The final data address should be controlled by stopping operations after the appropriate number of System clock pulses.

	TST BKP	NO MUX	FAST TRAN	LOAD DB	LOAD IO	MEMORY OFF	DB scan drive DB
normal mode	0	0	0	0	0	N	N
test of periph.bus keeper	1	0	0	0	0	N	N
memory dump	0	0	1	0	0	N	N
parallel memory dump	0	0	1	0	1	N	N
serial memory load	0	0	1	1	0	N	Y
serial data/instruct. for CPU	0	0	0	1	0	Y	Y

Note: all other bit configurations are forbidden.

MEMORY DUMP PROCEDURE (FASTRAN * LOADDB/ * LOADIO/):

1- Select a test mode (refer to “ST7 OPERATING MODES”). Normally, the ROMDUMP mode is selected.

2- The CPU executes a short program to initialise the CRC and to jump to the last address of the memory under test (FFFFh). These instructions can be provided by the test ROM or by the STC.

3- Enable the STC (V_{PP}_TEST high) and suspension of the System clock.

4- In MODER scan register, FASTRAN-bit is set via the STC to configure the CPU in dump mode.

5- Restart of the system clock for an appropriate number of pulses. The CPU is forced to execute a dedicated instruction and the memory is dumped (1 clock cycle per byte). During this phase, the address and data buses are accessible externally and the CRC analyses the data on DB bus.

6- Suspension of the System clock.

7- If there is another block of memory to test, the STC provides a jump instruction to the CPU and the test continues at step 4. If not, go to step 8.

8- Selection of the CRC via the SELSCR scan chain.

9- Shifting out and comparison of the CRC result.

PARALLEL MEMORY LOAD PROCEDURE (FASTRAN * LOADDB/ * LOADIO):

1- Select a test mode (refer to “ST7 OPERATING MODES”).

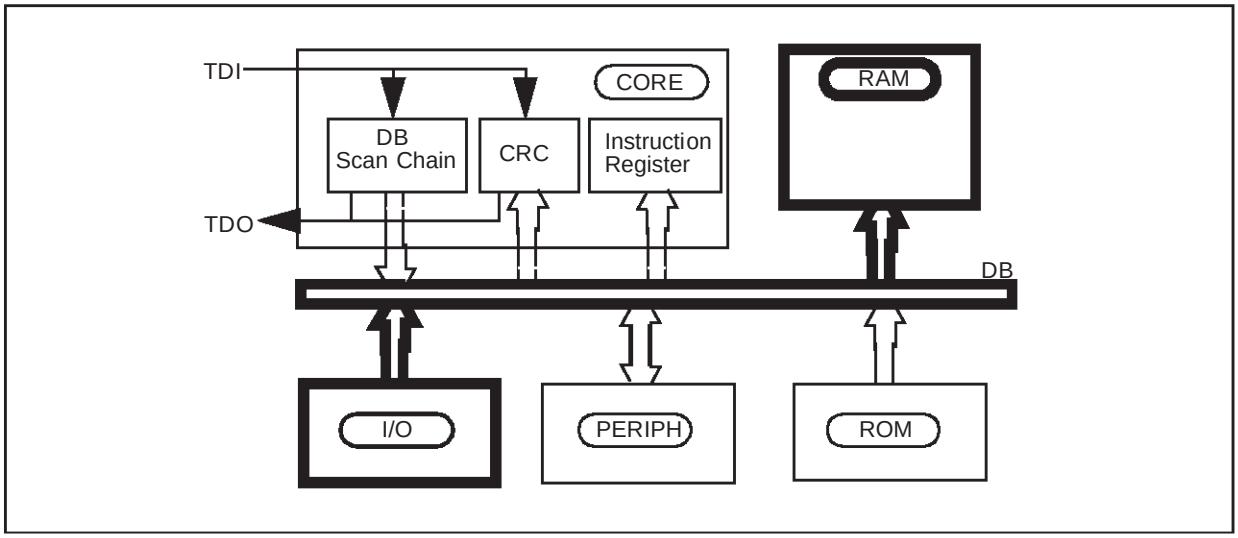
2- The CPU executes a short program to jump to the address where the last data will be loaded. These instructions can be provided by the test ROM or by the STC.

3- Enable the STC (V_{PP}_TEST high) and suspension of the System clock.

4- In MODER scan register
- FASTRAN-bit is set via the STC to configure the CPU in fast transfer mode.
- LOADIO-bit is set via the STC to configure the I/O port

5- Restart of the system clock for an appropriate number of pulses. The CPU is forced to execute a dedicated instruction. During this phase RW signal is low but the CPU does not drive the data bus. The data is provided by an I/O port. One byte is loaded in the memory on each clock cycle.

Figure 4. Parallel memory load with STC



SERIAL MEMORY LOAD PROCEDURE (FASTRAN * LOADDB * LOADIO/):

1- Select a test mode (refer to “ST7 OPERATING MODES”).

2- The CPU executes a short program to jump to the address where the last data will be loaded. These instructions can be provided by the test ROM or by the STC.

3- Enable the STC (V_{PP}_TEST high) and suspension of the System clock.

4- In MODER scan register
- FASTRAN-bit is set via the STC to configure the CPU in fast transfer mode.
- LOADDB-bit is set via the STC to configure the DB scan chain and to disable the ROM

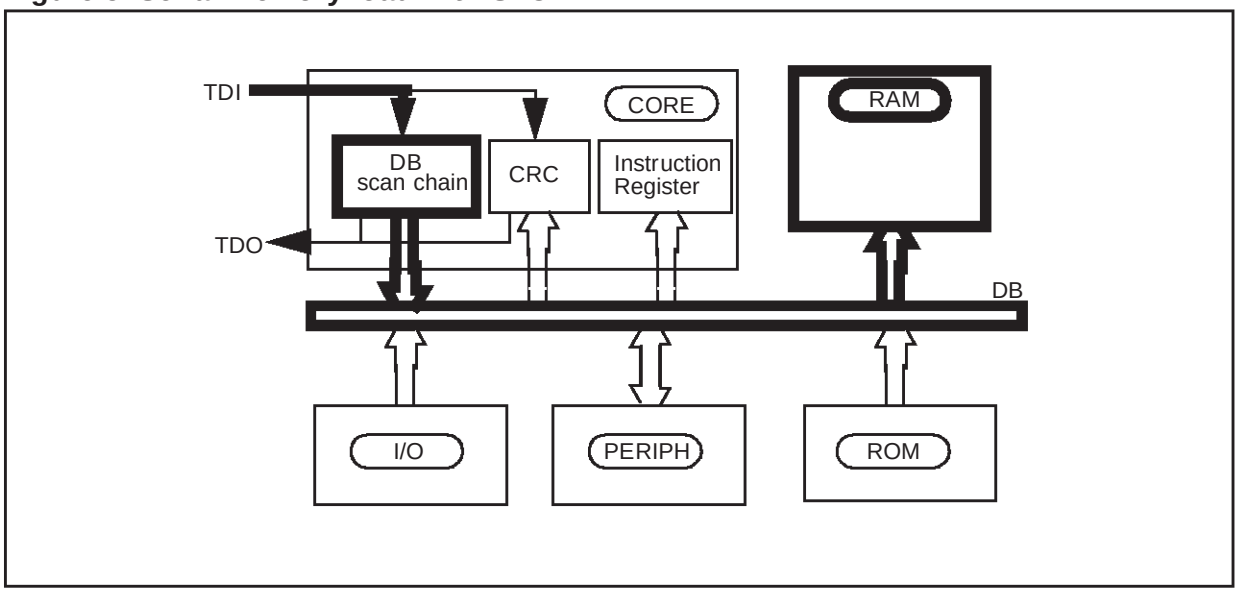
5- Selection of the DB scan chain via the SELSCR.

6- Shifting the data byte in DB.

7- Restart of the system clock for 1 pulse. The CPU is forced to execute a dedicated instruction. During this phase RW signal is low but the CPU does not drive the data bus. The data is provided by DB scan chain. One byte is loaded in the memory on each system clock cycle.

8- Suspension of the System clock and loop on step 6 for the next byte

Figure 5. Serial memory load with STC



Note: In this mode, the DB scan chain drives the data bus on RW * PHI1

3 ST7 OPERATING MODES

The ST7 can be used in one of the 8 following operating modes:

- USER mode
- five test modes:
  - CPUTEST
  - ROMDUMP
  - SELFTEST
  - OPEN
  - ROUTRAM
- EMULATION mode
- BOOTSTRAP mode

For the purpose of EPROM programming, only SELFTEST mode is mandatory if the JTAG interface is used. BOOTSTRAP mode can be used to implement a parallel programming interface. ROMDUMP mode can be used to perform a dump of the EPROM.

The selection of the mode is done at reset, according to the following table:

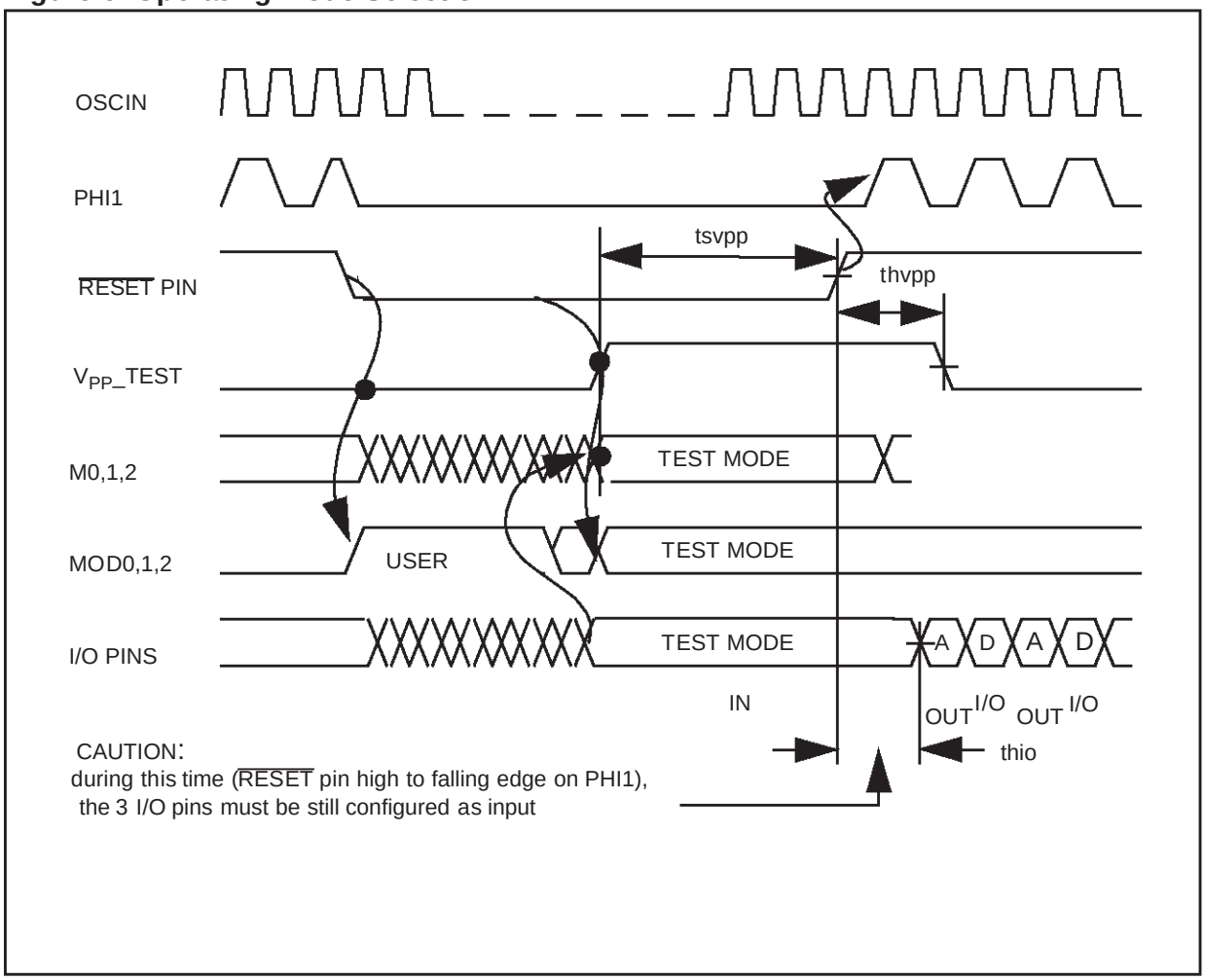
MODES	V_{PP}/TEST	M2	M1	M0
USER	GND	X	X	X
OPEN	V_{DD}	0	0	0
ROUTRAM	V_{DD}	0	1	1
ROMDUMP	V_{DD}	0	1	0
SELFTEST	V_{DD}	0	1	1
EMULATION	V_{DD}	1	0	0
CPUTEST	V_{DD}	1	0	1
BOOTSTRAP	V_{DD}	1	1	0

3.1 TIMING DIAGRAM FOR TEST MODE SELECTION

The test mode selection signals must respect the setup and hold times given in the diagram on next page:
tsvpp (min) = xxxnS (See Appendix)
thvpp (min) = yyynS (See Appendix)

All the other timings in the diagram are internal to the chip and therefore do not concern this document. Refer also to Appendix 10 - Timing Diagrams.

Figure 6. Operating Mode Selection



M0,M1,M2:

These 3 inputs normally come from 3 I/O pins. Their values are latched on the rising edge of NRESPAD (equivalent to RESET/ pin). When V_{PP}_TEST is high, they allow you to select the operating mode.

– V_{PP}_TEST:

This pin is mandatory for all ST7 devices. It has several functions:

– DURING RESET:

It allows to select the operating mode.

– IN USER MODE:

To select USER mode, it must be wired to GND.

If the Serial Test Controller is available in USER mode, it is activated when V_{PP}/TEST pin is high.

If the Serial Test Controller is available in USER mode, it is frozen when V_{PP}/TEST pin is low

(standard value).
V_{PP}/TEST pin must be directly wired to GND in the customer application.

– IN SELFTEST TEST MODES
If V_{PP}/TEST pin is low the Serial Test Controller is frozen.
If V_{PP}/TEST pin is high, the Serial Test Controller is enabled.
The operating mode is defined by M0, M1 and M2 inputs.

– IN EPROM VERSION:
V_{PP}/TEST pin is the high voltage power supply of the EPROM. This pin must be kept at about 12.5V during the EPROM programming sequence.

After a RESET or when exiting HALT mode, a temporisation is applied to wait for the oscillator stabilisation or RESET state recovery. It is possible to define the duration between 16 and 8192 CPU clock cycles (possible values are 16, 32, 64, 128, 256, 512, 1024, 2048, 4096 or 8192 CPU clock cycles).

This option is not active in SELFTEST. In these cases, a 16-cycle temporisation is applied to reduce the test duration.

3.2 SELFTEST

3.2.1 Purpose

This operating mode is used to test the digital cells, the I/O ports and some special functions (such interrupts) using a selftest routine previously loaded in RAM. This mode may also be used to program the EPROM and the EEPROM for industrial test. Compared to the standard EPROM programming mode (BOOTSTRAP), it allows to slightly reduce the test pattern length because the program is inside the device.

In SELFTEST mode, a 16-cycle temporisation is applied after reset or when the CPU exit from HALT mode.

ADDRESS MAPPING

- Interrupt and reset vectors are in test ROM:
$XYE0-$XYFF with XY=00/01/03/07/0F/1F/3F/7F defined by hardware

The addresses corresponding to the interrupt vectors are in RAM; the address corresponding to the reset vector is in the test ROM.

- All the features are internal.

- No external space.

I/O PORTS CONFIGURATIONS

Configured as in USER mode.

TEST SEQUENCE

1- Select SELFTEST mode during reset (V_{PP}_TEST * M2/ * M1 * M0)
2- The CPU starts to execute the program in test ROM
3- execution of the “loader” in test ROM. This program allows you to load the selftest routine into RAM. The data are read from an I/O port.
4- The CPU executes the “prebody” in test ROM. This short program initialises the CRC and all the features which are not initialised by reset. It also enables the STC.
5- Jump in RAM
6- Execution of the selftest routine loaded during step 3
7- Suspension of the System clock.
8- Selection of the CRC via the SELSCR scan chain.
9- Shifting out and comparison of the CRC result.

3.2.2 SELFTEST mode selection

The operating mode is selected during RESET phase.

In order to guarantee good synchronisation of the test patterns, the non-user modes must be selected on an external reset (and not on a power-on reset).

SELFTEST mode selection:
M0 = 1
M1 = 1
M2 = 0
V_{PP}/TEST = V_{DD}

3.3 ROMDUMP

3.3.1 Purpose

This mode is used to check all the internal RAM and ROM/EPROM content by executing a specific instruction. During this test, all the internal RAM and ROM/EPROM addresses are read and the corresponding data are analysed by the CRC. The memory dump can be done by software (for the RAM only) or by hardware. The hardware dump is performed when the core is configured in fast transfer mode (refer to MODER scan register). In this case, a byte is read on each clock cycle. In ROMDUMP mode, a 16-cycle temporisation is applied after reset or when the CPU exit from HALT mode. TRAP instruction is not available.

Address Mapping

- Reset vector is in test ROM:
 $XYDE-$XYDF with XY=00/01/03/07/0F/1F/3F/7F defined by hardware

- There is no interrupt vector

- All the features are internal.

- The I/O ports used to output control signals are not reachable by software.

- No external space

I/O Ports Configurations

- 8 I/O pins are configured according to the NOMUX signal coming from the Serial Test Controller:

- NOMUX = 0: the 8 I/O pins are used to output the data bus (on PHI1) and the 8 LSB of address bus (on PHI1).

- NOMUX = 1: the 8 I/O pins are used to output the data bus only.

In both cases, Data are always defined as output.

- 1 I/O pin is needed to output PHI1

- If 8 other I/O pins are available, they could be used to output the 8 MSB addresses.

Note: In ROMDUMP mode, I/O ports are configured as in CPUTEST mode.

ROMDUMP Program

This program may only be used to test the RAM. It is located in the test ROM. In most of the cases, the RAM is tested with the Serial Test Controller.

Test Sequence

1- Select ROMDUMP mode during reset (V_{PP}_TEST * M2/ * M1 * M0/)

2- The CPU execute a short program to initialise the CRC and to jump to the first address of the memory under test. These instructions can be provided by the test ROM or by the STC.

3- Enable the STC and suspension of the System clock.

4- In the MODER scan register, the FASTRAN and RWTRAN bits are set via the STC to configure the CPU in dump mode.

5- Restart of the system clock for an appropriate number of pulses. The CPU is forced to execute a dedicated instruction and the memory is dumped (1 clock cycle per byte). During this phase, the address and data buses are accessible externally and the CRC analyses the data on the DB bus.

6- Suspension of the System clock.

7- If there is another block of memory to test, the STC provides a jump instruction to the CPU and the test continues at step 5. If not, go to step 8.

8- Selection of the CRC via the SELSCR scan chain.

9- Shifting out and comparison of the CRC result. For the ROM, the CRC value does not depend on the customer code.

3.3.2 ROMDUMP mode selection

The operating mode is selected during RESET phase.

In order to guarantee a good synchronisation of the test patterns, the non-user modes must be selected on an external reset (and not on a power-on reset).

ROMDUMP mode selection:
M0 = 0
M1 = 1
M2 = 0
V_{PP}/TEST = V_{DD}

3.4 BOOTSTRAP

3.4.1 Purpose

This mode is used to program the internal EPROM by executing an external routine. Data are read in the external space and copied into the internal EPROM.

ADDRESS MAPPING

- Interrupt and reset vectors are in external space:

$XYE0-$XYFF with XY=00/01/03/07/0F/1F/3F/7F defined by hardware. Refer to Appendix 2.

- The I/O ports used for control signals are not reachable by software.

- All others features including the EPROM are internal.

- There is an external space (minimum size = 512 bytes). It corresponds to the test ROM area.

I/O PORTS CONFIGURATION

- 8 I/O pins are configured according to the NOMUX signal coming from the Serial Test Controller:

  - NOMUX = 0: the 8 I/O pins are used to output the data bus (on PHI1) and the 8 LSB of address bus (on PHI1).
  - NOMUX = 1: the 8 I/O pins are used to output the data bus only.

In both cases the pins are always defined as outputs except during a read of the external space.

- If 8 I/O pins are available, they could be used to output the 8 MSB addresses. This allows you to increase the external space size. At least one pin is needed to have a 512 bytes external space.

- 1 I/O pin is used to output PHI1 (= external clock)

- 1 I/O pin is used to output RW:
  - RW HIGH = read
  - RW LOW = write

- 1 I/O pin is used to output NESEL:
  - NESEL HIGH = internal access
  - NESEL LOW = external access

Refer to Appendix 10 for timing diagrams

Note: In BOOTSTRAP mode, the I/O ports are configured as in OPEN mode.

TEST SEQUENCE:

1- Select BOOTSTRAP mode during reset (V_{PP}_TEST * M2 * M1 * M0)

2- The CPU directly accesses to the external space and starts to execute an external program

3.4.2 BOOTSTRAP mode selection

The operating mode is selected during the RESET phase.

In order to guarantee a good synchronisation of the test patterns, the non-user modes must be selected on an external reset (and not on a power-on reset).

BOOTSTRAP mode selection:
M0 = 0
M1 = 1
M2 = 1
V_{PP}/TEST = V_{DD}

4 INTERNAL RAM LOADING VIA JTAG INTERFACE

The loading of the program in RAM is done using the JTAG interface and the operation is executed in serial mode.

The minimum pins used are:

- System pins: OSCIN, RESETN

- 5 pins JTAG interface

- 3 pins to force the test modes: M0, M1, M2 and V_{PP}/TEST

The internal RAM loading of the ST7 is done following these sequences:

1) Enter in the SELFTEST mode.

-This mode is selected during the RESET phase on the rising edge of RESET pin and applying a high voltage on V_{PP}/TEST pin. The operating mode is defined by M0, M1 and M2 inputs.

- The Serial Test Controller is enabled (V_{PP}_TEST high)

2) The OSCIN clock is held high before release the RESET signal

3) JTAG Initialisation

- MODER register is initialised to configure the clock: bit11 CKDIV2 = 1
  2 OSCIN = 1 system clock PHI1

4) Force a JUMP <address> instruction for the CPU to execute a short program to jump to the address where the last data will be loaded.
This address is calculated by adding the size of the code in bytes (maximum 256) with the RAMSTART address which is the address where the ST7 jumps at the end of the SELFTEST mode execution.

- Select DB scan chain and shift the JUMP instruction (CCh) in DataBus
- In MODER scan register
  set LOADDB bit (bit1) to configure the DB scan
  set CKDIV2 bit (bit11) to configure the internal clock
- 47 OSCIN clock is forced (23.5 PHI1)
  32 + 4 OSCIN clock (16 +2 PHI1) is forced for the RESET state recovery
  10 OSCIN clock (5 PHI1) to force the CPU to execute the instruction
  1 OSCIN clock (0.5 PHI1 to set it at a high level)) for the restart of system clock
- Select DB scan chain and shift the <address> in DataBus
  2 OSCIN clock after each data shift to force the CPU to execute the instruction

5) Load the program code in the internal RAM

- In MODER scan register:
  set FASTRAN bit (bit2) to configure the CPU in fast transfer mode
  set LOADDB bit (bit1) to configure the DB scan

set CKDIV2 bit (bit11) to configure the internal clock
- Select the DB chain
- Shift the DATA byte in DB

The first DATA shifted is the last data of the program to be loaded in RAM.
The address pointed by the CPU is decremented.
2 OSCIN clock is forced for the CPU execute the instruction.
The data are shifted until the first data of the program is placed at the RAMSTART address.

6) Internal RAM program execution
- Reset the ST7 with SELFTEST mode selection
- Free OSCIN clock
- The CPU executes the internal BOOTROM and jumps at the RAMSTART address to execute the program loaded in the internal RAM. All the I/O ports are accessible.

7) User assembler file
In the programming assembler file loaded in ram, the user has to configure some I/O Ports to get the data to be programmed from the external controller. The external controller could be the PC using the parallel port which will provide the data to the ST7.
The EPROM cell is programmed by executing a WFI instruction. The programming is ended with an interrupt to exit from WAIT mode. The interrupt vector is located at a specific address in the RAM (see tables in Appendix).

See in Appendix 6 an example of a programming assembler file.

4.1 SERIAL INSTRUCTIONS

This paragraph describes all the steps of the JTAG instructions for the internal RAM loading. All the JTAG sequences needed are detailed in Appendix 1: ”JTAG sequences”

Initialisation:
M0, M1, M2 are configured to select the Selftest Mode.
OSCIN are frozen before sending the JTAG sequence.
Reset
Test logic reset state

Configuring CPU General Purpose Test Control bits:
- select MODE instruction (6h = 0110)
- shift MODER value <11:0> (800h = 1000 0000 0000)

CKDIV2 = 1 (PHI1 = Oscin/2)

Data Bus Chain Selection:
- select SELSCAN instruction (8h = 1000)
- select DB chain (Ah = 1010)

- select INTSCAN instruction (4h = 0100)
- shift JUMP operation (CCh = 1100 1100)

Execute Instruction from Data Bus:

- select MODE instruction (6h = 0110)
- shift MODER value <11:0> (802h = 1000 0000 0010)

CKDIV2=1, LOADDB=1

- 47 OSCIN pulses (23,5 Pulses of PHI1)

Note: At this point reset state recovery is reached.

Data Bus Chain Selection:

- select SELSCAN instruction (8h = 1000)
- select DB chain (Ah = 1010)
- select INTSCAN instruction (4h = 0100)

Downloading High Address of bootstrap program in RAM:

- shift MSB byte of RAM Address to jump (<7:0>)
- 2 OSCIN pulses
- shift LSB byte of RAM Address to jump (<7:0>)
- 2 OSCIN pulses

Configuring CPU in Fast Transfer Mode and disabling the ROM:

- run test Idle state
- select MODE instruction (6h = 0110)
- shift MODER value <11:0> (806h = 1000 0000 0110)

CKDIV2=1, LOADDB=1, FASTRAN=1

Data Bus Chain Selection:

- select SELSCAN instruction (8h = 1000)
- select DataBus chain (Ah = 1010)
- select INTSCAN instruction (4h = 0100)
- send data RAM (256 th data <7:0>)
- 2 OSCIN pulses
- send data RAM (255 th <7:0>)
- 2 OSCIN pulses
- send data RAM (... <7:0>)
- 2 OSCIN pulses
- send data RAM (... <7:0>)
- 2 OSCIN pulses
- send data RAM (2 nd <7:0>)
- 2 OSCIN pulses
- send data RAM (1 rst <7:0>)

- 2 OSCIN pulses

At this point, the last data has been loaded at the RAMSTART address.

Internal RAM program execution:

M0,M1,M2 are configured to select the Selftest Mode

Reset the MCU

Test logic reset state (Reset the STC)

Configuring CPU General Purpose Test Control bits:

- select MODE instruction (6h = 0110)
- shift MODER value <11:0> (800h = 1000 0000 0000)

CKDIV2 = 1 (PHI1 = Oscin/2)

Free the OSCIN clock for the internal RAM program execution

5 EPROM HANDLING

The ST7 EPROM includes two control registers. These registers are not accessible in USER and EMULATION modes. The EPROM test and programming operations are performed with the following steps:

- select an operating mode using MOD0,1,2 and V_{PP}_TEST signals;
- configure the EPROM through the EPROM control registers;
- run the test or programming sequence.

5.1 EPROM CONTROL REGISTERS

EPCTRL2 Reset Value: 0000 0000, all flags are inactive

7							0
RES	RES	DMM	RES	RES	RES	ROTPRW	RES

EPCTRL1 Reset Value: 0000 0000, all flags are inactive

7							0
--	--	--	--	--	--	RES	RES

RES : Reserved

-- : Not implemented

DMM : **D**igital **M**argin **M**ode

ROTPRW : **R**ow **OTP R**ead **W**rite Enable: used to program the option byte

When the chip select for the control registers is active, address bit 0 allows to access to EPCTRL2 or EPCTRL1. The addresses of the EPROM control registers are in Appendixes 3 to 5.

5.2 PROGRAMMING OPERATION

A byte written at an EPROM address is programmed in the EPROM during WAIT mode. WAIT mode is entered by executing the WFI instruction and is exited by an interrupt. A timer can be used to accurately control the length of the programming pulse applied in this way.

Summary of a byte programming sequence (refer also to the flow chart in chapter 1.4)

1- Select SELFTEST or BOOTSRAP mode during reset

2- Write a data in an EPROM address (address and data are stored in EPROM latches).

3- Program the EPROM by executing a WFI instruction (the latch content is then copied in the EPROM array). Programming is done only if a write in an EPROM location was previously done.

4- End of programming with an interrupt to exit from WAIT mode.

5- Set DMM-bit. Read the programmed data to check that it was properly done. Reset DMM-bit. If the data is not well programmed, loop to 3.

Note: Between step 2 and 3 the CPU must not access the test ROM nor the EPROM.

An example of an assembler program that implements this algorithm is given in appendix 6. In this example the DMM bit is not set for verification, so an extra programming pulse is required for safety after the byte has been programmed. This way of programming is nevertheless not as it is recommended and more time consuming (extra pulse).

5.3 OPTION BYTE

Some members of the family (like the ST72T331XX, see Appendix 11 for the complete list) contain an option byte. When present, the option byte is structured as follows:

7							0
--	--	--	--	PCK1	PCK0	--	WDG

Bit [7:4] = Not Used

Bit3 = **PCK1** *PaCKage configuration 1*:
If set to 1 PF3, PF5, PE2 and PE3 IOs are enabled (QFP64 package).
If set to 0 PF3, PF5, PE2 and PE3 IOs are disabled (SDIP56, QFP44 and SDIP42 packages).

Bit2 = **PCK2** *PaCKage configuration 2*:
If set to 1 PA0, PA1, PA2, PB5, PB6, PB7,PD6, PD7, PE4, PE5, PE6 and PE7 IOs are enabled (QFP64 and SDIP56 packages).
If set to 0 PA0, PA1, PA2, PB5, PB6, PB7,PD6, PD7, PE4, PE5, PE6 and PE7 are disabled (QFP44 and SDIP42 packages).

Bit1 = Not Used

Bit0 = **WDG** *WDG Enable*
If set to 1 the Watchdog is not enabled after reset (Software Watchdog).
If set to 0 the Watchdog is enabled after reset (Hardware Watchdog).

The bits PCK0 and PCK1 must therefore be programmed according to the package, while the bit WDG can be programmed according to user needs. The following table summarizes the programming of the option byte:

PCK1	PCK0	WDG	CONFIGURATION
1	1	1	QFP64 - SW watchdog (1)
1	1	0	QFP64 - HW watchdog
0	1	1	SDIP56 - SW watchdog
0	1	0	SDIP56 - HW watchdog
0	0	1	QFP44 and SDIP42 - SW watchdog
0	0	0	QFP44 and SDIP42 - HW watchdog

(1) Default configuration after UV erase

5.4 PROGRAMMING OF THE OPTION BYTE

In order to program the option byte you must set the ROTPRW bit in the EPCTRL2 register. When this bit is set, the first row of the EPROM matrix is disabled, and a special row, called the OTP row, is accessed instead. The option byte is the first byte of this row, and it must be programmed following the same algorithm as for the rest of the memory. All the bytes of the OTP row except the first (option byte), are reserved for test purposes and must not be accessed.

6 APPENDIX 1: JTAG SEQUENCES

This chapter details the sequences of signals to apply to the JTAG interface to perform most of the operations described in this document. In the right column the state of the JTAG state-machine is reported (refer to Figure 1 - page 7)

- Run Test Idle

TRST = 1, TMS = 0, TDI = 1		(Run-Test-idle)
1 TCK pulse		

- Select MODE instruction (06h = 0110)

TMS = 1	1 TCK pulse	(select SR scan)
TMS = 1	1 TCK pulse	(select IR scan)
TMS = 0	1 TCK pulse	(capture IR)
TMS = 0	1 TCK pulse	(shift IR)
TDI = 0	1 TCK pulse	
TDI = 1	1 TCK pulse	
TDI = 1	1 TCK pulse	
TDI = 0		
TMS = 1	1 TCK pulse	(exit1 IR)
TMS = 1	1 TCK pulse	(update IR)

- Select SELSCAN instruction (8h = 1000)

TMS = 1	1 TCK pulse	(select SR scan)
TMS = 1	1 TCK pulse	(select IR scan)
TMS = 0	1 TCK pulse	(capture IR)
TMS = 0	1 TCK pulse	(shift IR)
TDI = 0	1 TCK pulse	
TDI = 0	1 TCK pulse	
TDI = 0	1 TCK pulse	
TDI = 1		
TMS = 1	1 TCK pulse	(exit1 IR)
TMS = 1	1 TCK pulse	(update IR)

- select INTSCAN instruction (4h = 0100)

TMS = 1	1 TCK pulse	(select DR scan)
TMS = 1	1 TCK pulse	(select IR scan)
TMS = 0	1 TCK pulse	(capture IR)
TMS = 0	1 TCK pulse	(shift IR)
TDI = 0	1 TCK pulse	
TDI = 0	1 TCK pulse	
TDI = 1	1 TCK pulse	
TDI = 0		

TMS = 1	1 TCK pulse	(exit1 IR)
TMS = 1	1 TCK pulse	(update IR)

- Select DataBus chain (Ah = 1010)

TMS = 1	1 TCK pulse	(select SR scan)
TMS = 0	1 TCK pulse	(capture SR)
TMS = 0	1 TCK pulse	(shift SR)
TDI = 0	1 TCK pulse	
TDI = 1	1 TCK pulse	
TDI = 0	1 TCK pulse	
TDI = 1		
TMS = 1	1 TCK pulse	(exit1 SR)
TMS = 1	1 TCK pulse	(update SR)

- Shift JUMP operation <7:0>: (CCh = 1100 1100)

TMS = 1	1 TCK pulse	(select SR scan)
TMS = 0	1 TCK pulse	(capture SR)
TMS = 0	1 TCK pulse	(shift SR)
TDI = bit0	1 TCK pulse	
TDI = bit1	1 TCK pulse	
TDI = bit2	1 TCK pulse	
....	.....	
....	.....	
TDI = bit6	1 TCK pulse	
TDI = bit7		
TMS = 1	1 TCK pulse	(exit1 SR)
TMS = 1	1 TCK pulse	(update SR)

- Shift MODER value <11:0>: xxxx xxxx xxxx

TMS = 1	1 TCK pulse	(select SR scan)
TMS = 0	1 TCK pulse	(capture SR)
TMS = 0	1 TCK pulse	(shift SR)
TDI = bit0	1 TCK pulse	
TDI = bit1	1 TCK pulse	
TDI = bit2	1 TCK pulse	
....	.....	
....	.....	
TDI = bit10	1 TCK pulse	
TDI = bit11		
TMS = 1	1 TCK pulse	(exit1 SR)
TMS = 1	1 TCK pulse	(update SR)

- Shift LOW/HIGH Nibble of Ram Address to jump (<7:0>)

TMS = 1	1 TCK pulse	(select SRscan)
TMS = 0	1 TCK pulse	(capture SR)
TMS = 0	1 TCK pulse	(shift SR)
TDI = bit0	1 TCK pulse	
TDI = bit1	1 TCK pulse	
TDI = bit2	1 TCK pulse	
....	.....	
....	.....	
TDI = bit6	1 TCK pulse	
TDI = bit7		
TMS = 1	1 TCK pulse	(exit1 SR)
TMS = 1	1 TCK pulse	(update SR)

- Send data RAM (data <7:0>)

TMS = 1	1 TCK pulse	(select SRscan)
TMS = 0	1 TCK pulse	(capture SR)
TMS = 0	1 TCK pulse	(shift SR)
TDI = bit0	1 TCK pulse	
TDI = bit1	1 TCK pulse	
TDI = bit2	1 TCK pulse	
....	.....	
....	.....	
TDI = bit6	1 TCK pulse	
TDI = bit7		
TMS = 1	1 TCK pulse	(exit1 DR)
TMS = 1	1 TCK pulse	(update DR)

7 APPENDIX 2: ST72 SUB FAMILIES AND NAMING CONVENTIONS

You will find in the next three appendixes some details relative to the three ST72 sub-families (named after the representative device of each family): ST72E77, ST72E331, ST72E251. This division in families is **not user issue**, but just a matter of convenience for EPROM programmers producers, as the test mode in each family has some common characteristics.

Below is an overview of:
- Address mapping Vs Test Mode;
- Pin assignment Vs Test Mode.

The values in these tables are different for the different sub-families and are detailed in Appendixes three to five.

7.1 TYPICAL ADDRESS MAPPING IN ST7 OPERATING MODES

<table>
<tr><th></th><th>USER</th><th>SELFTEST</th><th>OPEN</th><th>BOOTSTRAP</th></tr>
<tr><td></td><td>I/O register</td><td>I/O register</td><td>I/O reg/extern</td><td>not used</td></tr>
<tr><td></td><td>Periph. register</td><td>Periph. register</td><td>Periph. reg/extern</td><td>Periph. register</td></tr>
<tr><td></td><td>RAM</td><td>RAM</td><td>RAM</td><td>RAM</td></tr>
<tr><td></td><td>EEPROM</td><td>EEPROM</td><td>EEPROM</td><td>EEPROM</td></tr>
<tr><td></td><td rowspan="4">not used</td><td>not used</td><td>not used</td><td></td></tr>
<tr><td>$XY00</td><td rowspan="2">TEST ROM</td><td rowspan="3">external</td><td>external.</td></tr>
<tr><td>$XYDE
$XYDF
(1)</td><td>BOOTSTRAP
vectors
(external)</td></tr>
<tr><td>$XYE0
$XYFF
(1)</td><td>reset vectors
in test ROM</td><td>EXTERN.</td></tr>
<tr><td></td><td>not used</td><td>not used</td><td>not used</td><td>not used</td></tr>
<tr><td></td><td>User ROM/EPROM</td><td>User ROM/EPROM</td><td>external</td><td>User EPROM</td></tr>
<tr><td>$FFE0
$FFFF</td><td>User vectors</td><td>OPEN vectors
(external)</td><td>EMULAT.vectors
(external)</td><td>User vectors</td></tr>
</table>

Note 1: XY is defined by family ST72E77, ST72E331, ST72E251.

7.2 EXAMPLE I/O PORT FUNCTION VERSUS OPERATING MODES

PAD	USER	SELFTEST	BOOTSTRAP
PX0	PX0	PX0/M0	A0/DB0/M0
PX1	PX1	PX1/M1	A1/DB1/M1
PX2	PX2	PX2/M2	A2/DB2/M2
PX3	PX3	PX3	A3/DB3
PX4	PX4	PX4	A4/DB4
PX5	PX5	PX5	A5/DB5
PX6	PX6	PX6	A6/DB7

PX7	PX7	PX7	A7/DB7
PY0	PY0	JTDO	A8
PY1	PY1	PY1/JTMS	A9
PY2	PY2	PY2/JTRST	A10
PY3	PY3	PY3/JTCK	A11
PY4	PY4	PY4/JTDI	A12
PY5	PY5	PY5	A13
PY6	PY6	PY6	A14
PY7	PY7	PY7	A15
PZ0	PZ0	PZ0	PHI1
PZ1	PZ1	PZ1	RW
PZ2	PZ2	PZ2	NESEL
PZ3	PZ3	PZ3	NARESET
PZ4	PZ4	PZ4	
PZ5	PZ5	PZ5	
PZ6	PZ6	PZ6	
PZ7	PZ7	PZ7	
V_{PP}/TEST	V_{PP}/TEST	V_{PP}/TEST	V_{PP}/TEST
RESETN	RESETN	RESETN	RESETN
OSCOUT	OSCOUT	OSCOUT	OSCOUT
OSCIN	OSCIN	OSCIN	OSCIN

7.3 NAMING CONVENTIONS

The ST72 family devices are named according to the following conventions:

Example: ST72 T 213 G 2 B 6

- ST72: family name
- T: T = OTP, E = EPROM, F = FLASH, No char = ROM
- 213: Sub family[1)] and Subset index (last digit not mandatory)
- G: User pin count
- 2: ROM size code (1=4K, 2=8K, 4=16K, 6=32K)
- B: Package (B=Plastic DIP, D=Ceramic DIP, M=Plastic SO, T=Plastic TQFP)
- 6: Temperature code

Note 1: The subfamily is not necessarily the same as the subfamily categories listed in Appendixes 3 to 5.

8 APPENDIX 3: GENERAL PURPOSE ST72E77

This appendix contains technical details about the ST72E77 sub-family. Devices of this sub-family have no option byte.

8.1 ST72E77 OVERVIEW

OTP EPROM	EPROM	EEPROM	RAM	Package	JTDI PA7	JTCK PA6	JTRST PA5	JTMS PA4	JTDO PC6	M2 DA3	M1 DA2	M0 DA1
ST72x272K4	16K C000-FFFF	-	512	SO34 SDIP32	29 27	30 28	31 29	32 30	25 24	3 3	2 2	1 1
OTP EPROM	**EPROM**	**EEPROM**	**RAM**	**Package**	**JTDI PA7**	**JTCK PA6**	**JTRST PA5**	**JTMS PA4**	**JTDO PC6**	**M2 DA2**	**M1 DA1**	**M0 DA0**
ST72x372J4	16K C000-FFFF	-	512	SDIP42	34	35	36	37	31	2	1	42
ST72x371N4	16K C000-FFFF	-	512	TQF64 SDIP56	28 46	29 47	30 48	31 49	20 38	43 3	42 2	41 1
ST72x671N6	16K C000-FFFF	-	1024	TQF64 SDIP56	28 46	29 47	30 48	31 49	20 38	43 3	42 2	41 1

8.2 INTERNAL RAM VECTORS IN SELFTEST MODE

Vectors	
RAM program first address RAMSTART	0070h
Timer TOF interrupt (RAMSTART + $C0 + $11)	0141h
Timer OCMP interrupt (RAMSTART + $C0 + $14)	0144h
Timer ICAP interrupt (RAMSTART + $C0 + $1D)	014Dh
ITA interrupt	0088h
ITB interrupt	0081h
ITC interrupt	007Ah
ITD interrupt	0073h

8.3 PIN OUT CONFIGURATION IN ST72E77 OPERATING MODES

USER	SELFTEST	ROMDUMP	BOOTSTRAP
DA 0..2	DA 0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset
DA 3..7	DA 3..7	A3..7/DB3..7	A3..7/DB3..7
DA8	DA8	A8	A8
PA0	PA0	-	-
PA1	PA1	-	NESEL
PA2	PA2	-	-
PA3	PA3	-	-
PA4	PA4/TMS (JTAG)	PA4/TMS (JTAG)	PA4/TMS (JTAG)
PA5	PA5/TRST (JTAG)	PA5/TRST (JTAG)	PA5/TRST (JTAG)
PA6	PA6/TCK (JTAG)	PA6/TCK (JTAG)	PA6/TCK (JTAG)

PA7	PA7/TDI (JTAG)	PA7/TDI (JTAG)	PA7/TDI (JTAG)
PC6	PC6/TDO (JTAG)	PC6/TDO (JTAG)	PC6/TDO (JTAG)
PB 1..4	PB 1..4	A10..13	A10..13
PB5	PB5	A15	A15
PB6	PB6	A14	A14
PB7	PB7	-	-
PD3	PD3	PHI1	PHI1
PD4	PD4	RW	RW
PD5	PD5	A9	A9 or ARESTN
V_{PP}	V_{PP}/TEST	V_{PP}/TEST	V_{PP}/TEST
Other pins	same	same	same

8.4 ADDRESS MAPPING IN ST72E77 OPERATING MODES

	USER	SELFTEST	ROMDUMP	BOOTSTRAP
$0000 $0001	PORT A registers	PORT A registers	not used	not used
$0002 $0003	PORT C registers	PORT C registers	PORT C registers	PORT C registers
$0004 $0005	PORT D registers	PORT D registers	not used	not used
$0006 $0007	PORT B registers	PORT B registers	not used	not used
$0008 $005F	PERIPH registers	PERIPH registers	PERIPH registers	PERIPH registers
$0057	Reserved	EPCTRL1	Reserved	EPCTRL1
$0058	Reserved	EPCTRL2	Reserved	EPCTRL2
$0060 $03FF	RAM	RAM	RAM	RAM
$0400 $7EFF	not used	not used	not used	not used
$7F00 $7FDD	not used	TEST ROM	TEST ROM	external
$7FDE $7FDF		ROMDUMP vectors in test ROM	ROMDUMP vectors in test ROM	
$7FE0 $7FFF		SELFTEST vectors in test ROM	SELFTEST vectors in test ROM	BOOTSTRAP vectors (external)
$8000 $FFDF	USER ROM /EPROM	USER ROM /EPROM	USER ROM /EPROM	USER ROM /EPROM
$FFEF $FFFF	USER vectors	USER vectors	USER vectors	USER vectors

9 APPENDIX 4: GENERAL PURPOSE ST72E331

This appendix provides technical details about the ST72E331 sub-family. Devices of this sub-family have one option byte.

9.1 ST72E331 OVERVIEW

OTP EPROM	EPROM	EEPROM	RAM	Package	JTDI PA7	JTCK PA6	JTRST PA5	JTMS PA4	JTDO PA3	M2 DA3	M1 DA2	M0 DA1
ST72x121J4	16K C000-FFFF	-	512	TQFP44	37	36	35	34	31	4	3	2
ST72x311J4				SDIP42	30	29	28	27	24	41	40	39
ST72x311N4	16K C000-FFFF	-	512	TQFP64	52	51	50	49	46	7	6	5
				SDIP56	40	39	38	37	34	55	54	53
ST72x331J4	16K C000-FFFF	256 C00-CFF	512	TQFP44	37	36	35	34	31	4	3	2
				SDIP42	30	29	28	27	24	41	40	39
ST72x331N4	16K C000-FFFF	256 C00-CFF	512	TQFP64	52	51	50	49	46	7	6	5
				SDIP56	40	39	38	37	34	55	54	53
ST72x511R6	32K 8000-FFFF	-	1024	TQFP64	52	51	50	49	46	7	6	5
ST72x531R6	32K 8000-FFFF	256 C00-CFF	1024									

9.2 INTERNAL RAM VECTORS IN SELFTEST MODE

Vectors	RAM address
RAM program first address RAMSTART (RAMBEG)	080h
Timer 1 interrupt (RAMBEG+$D0)	150h
Timer 2 interrupt (RAMBEG+$D0)	150h
ITA interrupt (RAMBEG+$10)	090h
ITB interrupt (RAMBEG+$20)	0A0h
ITC interrupt (RAMBEG+$30)	0B0h
ITD interrupt (RAMBEG+$40)	0C0h

9.3 PIN OUT CONFIGURATION IN ST72E331 OPERATING MODES

USER	SELFTEST	ROMDUMP	BOOTSTRAP
PB0.2	DA 0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset
PB3..7	DA 3..7	A3..7/DB3..7	A3..7/DB3..7
PD0	PD0	A9	A9
PD1..6	PD1..6	A10..15	A10..15
PA2	PA2		NESEL
PA3	PA3/TDO (JTAG)	PA3/TDO (JTAG)	PA3/TDO (JTAG)
PA4	PA4/TMS (JTAG)	PA4/TMS (JTAG)	PA4/TMS (JTAG)

PA5	PA5/TRST (JTAG)	PA5/TRST (JTAG)	PA5/TRST (JTAG)
PA6	PA6/TCK (JTAG)	PA6/TCK (JTAG)	PA6/TCK (JTAG)
PA7	PA7/TDI (JTAG)	PA7/TDI (JTAG)	PA7/TDI (JTAG)
PF0	PF0	PHI1	PHI1
PF1	PF1	RW	RW
PF2	PF2	A8	A8
V_{PP}	V_{PP}/TEST	V_{PP}/TEST	V_{PP}/TEST
Other pins	same	same	same

9.4 ADDRESS MAPPING IN ST72E331 OPERATING MODES

	USER	**SELFTEST**	**ROMDUMP**	**BOOTSTRAP**
$0000 $0003	PORT A registers	PORT A registers	PORT A registers	not used
$0004 $0007	PORT C registers	PORT C registers	PORT C registers	PORT C registers
$0008 $000B	PORT B registers	PORT B registers	PORT B registers	not used
$000C $000F	PORT E registers	PORT E registers	PORT E registers	PORT E registers
$0010 $0013	PORT D registers	PORT D registers	PORT D registers	not used
$0014 $0017	PORT F registers	PORT F registers	PORT F registers	not used
$0013 $007F	PERIPH registers	PERIPH registers	PERIPH registers	PERIPH registers
$002D	Reserved	EPCTRL1	Reserved	EPCTRL1
$002E	Reserved	EPCTRL2	Reserved	EPCTRL2
$0080 $027F	RAM	RAM	RAM	RAM
$0280 $0BFF	not used	not used	not used	not used
$0C00 $0CFF	EEPROM	EEPROM	EEPROM	EEPROM
$0D00 $BEFF	Reserved	TEST ROM	TEST ROM	external
$BF00 $BFDD		not used	not used	
$BFDE $FFDF		ROMDUMP vectors in test ROM	ROMDUMP vectors in test ROM	
$BFE0 $BFFF		SELFTEST vectors in test ROM	SELFTEST vectors in test ROM	BOOTSTRAP vectors (external)
$C000 $FFDF	USER ROM /EPROM	USER ROM /EPROM	USER ROM /EPROM	USER ROM /EPROM
$FFEF $FFFF	USER vectors	USER vectors	USER vectors	USER vectors

10 APPENDIX 5: GENERAL PURPOSE ST72E251

This appendix contains technical details about the ST72E251 sub-family. Devices of this sub-family have no option byte.

10.1 ST72E251 OVERVIEW

OTP EPROM	EPROM	EEPROM	RAM	Package	JTDI PA3	JTCK PA2	JTRST PA1	JTMS PA0	JTDO PC3	M2 PB2	M1 PB1	M0 PB0
ST72x102G2	8K E000-FFFF	-	256	PSO28	22	23	24	25	14	9	10	11
				SDIP32	26	27	28	29	16	11	12	13
ST72x212G2	8K E000-FFFF	-	256	PSO28	22	23	24	25	14	9	10	11
				SDIP32	26	27	28	29	16	11	12	13
ST72x251G2	8K E000-FFFF	-	256	PSO28	22	23	24	25	14	9	10	11
				SDIP32	26	27	28	29	16	11	12	13

10.2 INTERNAL RAM VECTORS IN SELFTEST MODE

Vectors	RAM address
RAM program first address RAMSTART (RAMBEG+$10)	090h
Timer 1 interrupt (RAMSTART+$C0)	150h
Timer 2 interrupt (RAMSTART+$C0)	150h
ITA interrupt (RAMSTART+$03)	093h
ITB interrupt (RAMSTART+$10)	0A0h
ITC interrupt (RAMSTART+$10)	0A0h

10.3 PIN OUT CONFIGURATION IN ST72E251 OPERATING MODES

USER	SELFTEST	ROMDUMP	BOOTSTRAP
PB0.2	PB 0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset	A0..2/DB0..2 M0..2 on Reset
PB3..7	PB 3..7	A3..7/DB3..7	A3..7/DB3..7
PA0	PA0/TMS (JTAG)	PA0/TMS (JTAG)	PA0/TMS (JTAG)
PA1	PA1/TRST (JTAG)	PA1/TRST (JTAG)	PA1/TRST (JTAG)
PA2	PA2/TCK (JTAG)	PA2/TCK (JTAG)	PA2/TCK (JTAG)
PA3	PA3/TDI (JTAG)	PA3/TDI (JTAG)	PA3/TDI (JTAG)
PA4..7	PA4..7	PA4..7	PA4..7
PC0	PC0	PHI1	PHI1
PC1	PC1	RW	RW
PC2	PC2	PC2	NESEL
PC3	PC3/TDO (JTAG)	PC3/TDO (JTAG)	PC3/TDO (JTAG)
V_{PP}	V_{PP}/TEST	V_{PP}/TEST	V_{PP}/TEST
Other pins	same	same	same

10.4 ADDRESS MAPPING IN ST72E251 OPERATING MODES

<table>
<tr><th></th><th>USER</th><th>SEFTEST</th><th>ROMDUMP</th><th>BOOTSTRAP</th></tr>
<tr><td>$0000
$0003</td><td>PORT C
registers</td><td>PORT C
registers</td><td rowspan="2">not used</td><td rowspan="2">not used</td></tr>
<tr><td>$0004
$0007</td><td>PORT B
registers</td><td>PORT B
registers</td></tr>
<tr><td>$0008
$000B</td><td>PORT A
registers</td><td>PORT A
registers</td><td>PORT A
registers</td><td rowspan="2">not used</td></tr>
<tr><td>$000C
$000F</td><td>not used</td><td>not used</td><td>not used</td></tr>
<tr><td>$0020
$007F</td><td>PERIPH
registers</td><td>PERIPH
registers</td><td>PERIPH
registers</td><td>PERIPH
registers</td></tr>
<tr><td>$0027</td><td>Reserved</td><td>EPCTRL1</td><td>Reserved</td><td>EPCTRL1</td></tr>
<tr><td>$0026</td><td>Reserved</td><td>EPCTRL2</td><td>Reserved</td><td>EPCTRL2</td></tr>
<tr><td>$0080
$017F</td><td>RAM</td><td>RAM</td><td>RAM</td><td>RAM</td></tr>
<tr><td>$0180
$DEFF</td><td>not used</td><td>not used</td><td>not used</td><td>not used</td></tr>
<tr><td>$DF00
$DFDD</td><td rowspan="3">Reserved</td><td>TEST ROM</td><td>TEST ROM</td><td rowspan="2">external</td></tr>
<tr><td>$DFDE
$DFDF</td><td>ROMDUMP
vectors in test ROM</td><td>ROMDUMP
vectors in test ROM</td></tr>
<tr><td>$DFE0
$DFFF</td><td>SELFTEST vectors
in test ROM</td><td>SELFTEST vectors
in test ROM</td><td>BOOTSTRAP
vectors (external)</td></tr>
<tr><td>$E000
$FFE3</td><td>USER ROM
/EPROM</td><td>USER ROM
/EPROM</td><td>USER ROM
/EPROM</td><td>USER ROM
/EPROM</td></tr>
<tr><td>$FFE4
$FFFF</td><td>USER
vectors</td><td>USER
vectors</td><td>USER
vectors</td><td>USER
vectors</td></tr>
</table>

11 APPENDIX 6: EXAMPLE OF PROGRAMMING ASSEMBLER FILE

This example is taken from the ST7 EPROM Programmer developed in ST laboratories.

In this example the EPROMER is driven by the PC through the parallel port. After loading the internal RAM, the pins used as JTAG signals are configured as I/Os.

-The data is provided by the PC bitwise (serially) on the I/O port bit PA6 (DATAIN).

-The handshake between the PC and the ST7 microcontroller is done by PA0 (PCFLAG) as an ST7 input and PA2 (ST7FLAG) as an ST7 output.

-The result of the programming is shifted out on PA3 as an output (DATAOUT)

After the first programming, the content of the EPROM address programmed is compared against the data to be programmed. If the data matches a second programming is done to secure the byte

programmed. If the comparison fails, five programming trials are done and the programming is stopped if the comparison always failed.

A version of this assembler program using the DMM feature is under development (see ”1.4 Programming Algorithm”)

```
;****************************
; Register definition    *
;****************************

DRA    .EQU   0000h           ;Porta data register
DDRA   .EQU   0001h           ;Porta data direction register
ORA    .EQU   0002h           ;porta option register(1 for push pull
                              ;with DDRA to 1)
E2PCR  .EQU   002Ch
TCR2   .EQU   0031h           ;Timer1 control register 2
TCR1   .EQU   0032h           ;Timer1 control register 1
TSR    .EQU   0033h           ; timer1 status register
OCHR1  .EQU   0036h           ; timer1 compare high register
OCLR1  .EQU   0037h           ; timer1 compare low register
CRH    .EQU   0038h           ; timer1 counter high register
```

```
CRL    .EQU   0039h          ; timer1 counter low register
CHRA   .EQU   003ah          ; timer1 alternate counter high
CLRA   .EQU   003bh          ; timer1 alternate counter low
OCHR2  .EQU   003eh          ; timer2 compare high register
OCLR2  .EQU   003fh          ; timer2 compare low register

RAMBEG.EQU    0080h          ; Beginning of RAM
FLAGS  .EQU   0081h          ; Flag of dtecet error
VY     .EQU   0082h          ; Number of trys
PROG_TRY.EQU   0083h         ;
MEMTMP.EQU    0084h          ; Data register
HIGHAD.EQU    0086h          ; MSB of EPROM address to be programmed
LOWAD  .EQU   0088h          ; LSB of EPROM address to be programmed
RAMSTART .EQU   0090h        ;Address of first instruction
                             ;out of Reset in selftest mode
RAMEND.EQU    017Fh          ;End of RAM

;*************************
;    PROGRAM       *
;    ST72Exx       *
*************************



BEGIN:.ORG  RAMSTART
      RSP                    ;Reset stack pointer
      LD    A,#00Eh          ;(TDI PA3,TDO PA2) as output
      LD     DDRA,A          ;
      LD     A,#00Eh         ;PA3-PA2 as push pull
      LD    ORA,A            ;
```

```
      NOP

PROG: LD    X,LOWAD
      CLR   FLAGS
      CLR   VY
      BSET  TCR2,#3           ;clock timer=Phi/8=CLOK ext/16
      CLR   TCR1              ;Timer reset
      LD    A,#070h           ;Mask interrupt on timer 2
      LD    OCHR2,A           ;
      BRES  DRA,#2            ;(TDO PA2)ST7FLAG Low
WAIT1A4:BTJTDRA,#0,WAIT1A4    ;(TMS PA0)waiting while PCFLAG High

ROM2: CALL  MEMWR             ; read the data to be programmed

WAITPRG:BTJFDRA,#0,WAITPRG    ;(TMS PA0)waiting while PCFLAG Low

BOUCTP:CALLEPROMP             ; first Programming
      LD    A,([HIGHAD.w],X)  ; read the EPROM address programmed
      CP    A,MEMTMP          ; compare the data read against the data to
be programmed
      JRNE   RESULTP

      CALLR  EPROMP           ; 2nd Programming
      LD    A,([HIGHAD.w],X)  ; read the EPROM address programmed
      CP    A,MEMTMP          ; compare the data read against the data to
be programmed
      JRNE   RESULTP

      CLR   VY
      JP    RESTP
```

```
DEVAM:INC   X                ; increment the LSB address
      CP    X,#00h
      JRNE  ROM2
      INC   HIGHAD           ; increment the MSB address
      JRA   ROM2

;*************************************************
;Transfer of data(Byte) MSB..LSB     *
;from PC to ST7 bit per bit        *
;Return value in MEMTMP location(Adress)  *
;*************************************************

MEMWR: LD   Y,#01h           ;
      CLR   MEMTMP
LECW: BRES  DRA,#2           ;(TDO PA2)ST7FLAG Low
WAIT2A4:BTJF  DRA,#0,WAIT2A4 ;(TMS PA0)waiting while PCFLAG Low
      BSET  DRA,#2           ;(TDO PA2)ST7FLAG High
WAIT3A4:BTJT  DRA,#0,WAIT3A4;(TMS PA0)waiting while PCFLAG High
      LD    A,DRA            ;(TCK PA6)read of portA
      AND   A,#080h          ;isolate DATAIN by masking portA
      JREQ  DAT0
      LD    A,#01h           ;If DATAIN NOT Nul A=1
      JP    DAT_IN
DAT0: LD    A,#00h           ;If DATAIN Nul A=0
DAT_IN:ADD   A,MEMTMP
      LD    MEMTMP,A
      SLA   Y                ;increment bit counter
      CP    Y,#00h
      JREQ  SORTW
      SLA   MEMTMP
```

```
       JRA   LECW
SORTW: BRES   DRA,#2           ;(TDO PA2)ST7FLAG Low
       RET

EPROMP:
       LD    A,MEMTMP
       CP    A,#0FFh
       JREQ  FINEPR
       LD    A,#00h           ; (160+4 timer cycle)
       LD    OCHR1,A          ; Mask data on pattern
       LD    A,#0FAh          ; (160+4 timer cycle)
       LD    Y,TSR            ; Mask data on pattern
       LD    OCLR1,A
       LD    A,MEMTMP
       LD    ([HIGHAD.w],X),A  ; WRITE THE LATCHED DATA TO EPROM
       BSET  DRA,#1
       CLR   CRL              ; RESET COUNTER
       BSET  TCR1,#6          ; ENABLE TIMER OCF INTERRUPT
       WFI                    ; TEMPO FOR PROGRAMMING(500us)
FINEPR: RET


RESULTP: INC   VY
       LD    Y,VY
       CP    Y,#5
       JRNE  BOUCTP
       BSET  FLAGS,#0         ; Number of trials equal 5
RESTP: CLR   Y
       BSET  DRA,#2           ; (TDO PA2)ST7FLAG High
WAIT4A4: BTJT  DRA,#0,WAIT4A4 ; (TMS PA0)waiting while PCFLAG High
```

```
      CP    Y,FLAGS
      JRNE  NIVH
      BRES  DRA,#3          ; (TDI PA3)DATAOUT Low
      JRA   VALPC6
NIVH: BSET  DRA,#3          ; (TDI PA3)DATAOUT High
VALPC6:NOP
      BRES  DRA,#2          ; (TDO PA2)ST7FLAG Low
WAIT5A5:BTJF  DRA,#0,WAIT5A5 ; (TMS PA0)waiting while PCFLAG Low
      JP    DEVAM
;*************************************************
;        Interrupt program    *
; used to allow delay for programming *
;        on EPROM location    *
;*************************************************
      .ORG  0150h           ; Timer Interrupt address in RAM
      NOP
      BRES  TCR1,#6         ;Interrupt inhibit
      IRET
```

12 APPENDIX 8: EXAMPLES OF JTAG WAVEFORMS

13 APPENDIX 9: ELECTRICAL CHARACTERISTICS

Input and output voltages for normal functions pins follow the data sheet of the device.

13.1 ABSOLUTE MAXIMUM RATINGS

Voltage on V_{DD} ..... 0.3 to 6V

Voltage on Reset/V_{PP} ..... 0.3 to 13V

Operating temperature ..... - 40°C to +125°C

13.2 DC CHARACTERISTICS

NAME	Min	Max	Unit	Conditions
V_{CC}	4.5	5.5	V	
I_{CC}		20	mA	8.66 MHz, 5.5V
FREQ		8.66	MHz	5V
V_{PP}	12.25	12.75	V	

13.3 RECOMMENDED VALUES IN PROGRAMMING MODE

During the programming, the power supply (V_{CC}), the programming voltage (V_{PP}) and the oscillator must respect the following conditions:

V_{CC} = 6.0 Volts

V_{PP} = 12.5 Volts

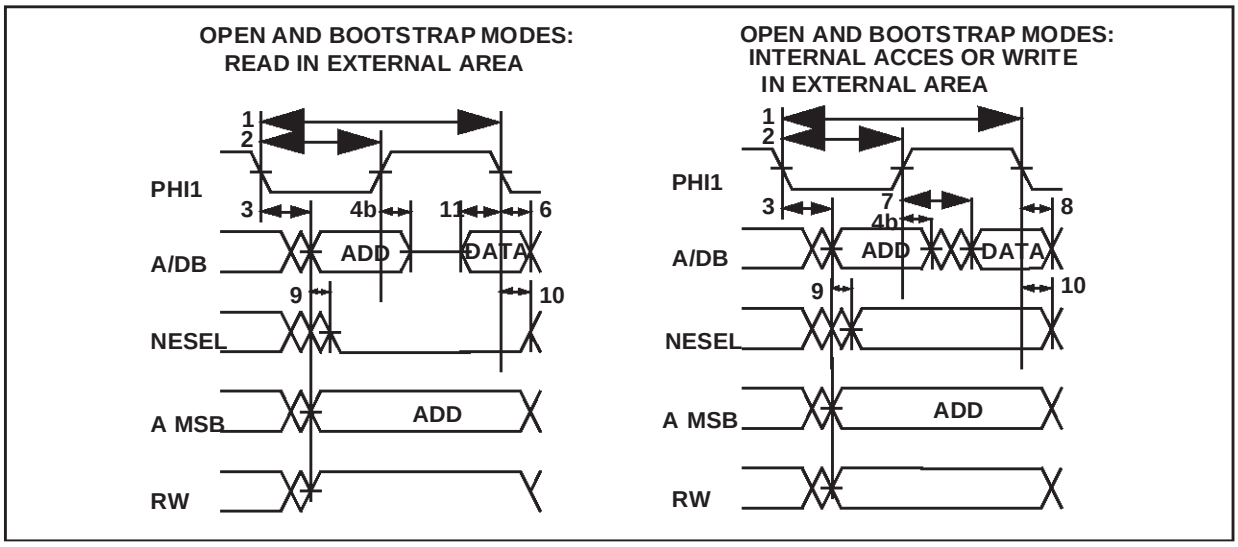
FREQ = 8 MHz

PROG. PULSE LENGTH = 20 µS

MAX PROG PULSES PER BYTE = 50 (typical 1 to 5)

14 APPENDIX 10: TIMING DIAGRAMS

Timings are evaluated by simulation for typical conditions (room temperature, V_{DD}=5V). Measured timings will be available after device characterization. JTAG timings will be available after device characterization.



	CHARACTERISTICS	MIN (ns)	MAX (ns)
1	Clock cycle time	125	dc
2	Clock width	60	
3	Address and RW delay time		25.5
4a	Address and RW hold time	4.50	
4b	PHI1 rising edge to HZ		4.5
5	PHI1 rising edge to active control signal		20
6	DATA hold time for external read	0	
7	PHI1 rising edge to output DATA		35
8	Output DATA hold time	5	
9	NESEL delay		10
10	NESEL hold time	0	
11	DATA read set up time	45	
tres	External reset pulse width	250	
tsvpp	V_{PP}/TEST setup time	250	
thvpp	V_{PP}/TEST hold time	50	
thio	I/O pin hold time for M0, M1, M2	20	

15 APPENDIX 11: LIST OF EXISTING DEVICES

This appendix lists the existing devices in OTP/EPROM version of the ST72 family that you can order with the request form attached to this document. This list is not exhaustive but allows you to get components able to emulate any other existing type.

SALES TYPE (3)	PACKAGE (4)	OPTION BYTE	READOUT PROTECTION	REPRESENTATIVE DEVICE SALES TYPE	REPRESENTATIVE DEVICE AVAILABILITY
ST72T101GxBy	SDIP32	None	None	ST72T251G2B6	5 samples sent with this spec
ST72T101GxMy	SO28	None	None		
ST72T213GxBy	SDIP32	None	None		
ST72T213GxMy	SO28	None	None		
ST72T212GxBy	SDIP32	None	None		
ST72T212GxMy	SO28	None	None		
ST72T251GxBy	SDIP32 / CSDIP32	None	None		
ST72T251GxMy	SO28	None	None		
ST72T121JxByz	SDIP42	1-Dynamic	None	ST72T331J4B6S	5 samples sent with this spec
ST72T121JxTyz	TQFP44	1-Dynamic	None		
ST72T311JxByz	SDIP42	1-Dynamic	None		
ST72T311JxTyz	TQFP44	1-Dynamic	None		
ST72T331JxByz	SDIP42	1-Dynamic	None		
ST72T331JxTyz	TQFP44	1-Dynamic	None		
ST72T311N2Byz	SDIP56	1-Dynamic	None	ST72T331N4B6S	5 samples sent with this spec
ST72T311N2Tyz	TQFP64	1-Dynamic	None		
ST72T311N4Byz	SDIP56	1-Dynamic	None		
ST72T311N4Tyz	TQFP64	1-Dynamic	None		
ST72T331N2Byz	SDIP56	1-Dynamic	None		
ST72T331N2Tyz	TQFP64	1-Dynamic	None		
ST72T331N4Byz	SDIP56	1-Dynamic	None		
ST72T331N4Tyz	TQFP64	1-Dynamic	None		
ST72T311N6By	SDIP56	None	None	ST72T531R6T6	5 samples sent with this spec
ST72T311N6Ty	TQFP64	None	None		
ST72T331N6By	SDIP56	None	None		
ST72T331N6Ty	TQFP64	None	None		
ST72T511RxTy	TQFP64	None	None		
ST72T531RxTy	TQFP64	None	None		

ST72T272KxBy ST72T272KxMy	SDIP32 SO34	None None	None None	ST72T272K4B6	5 samples sent with this spec
ST72T372J4By	SDIP42	None	None	ST72T372J4B6	5 samples sent with this spec
ST72T371NxBy ST72T371NxTy ST72T671NxBy ST72T671NxTy	SDIP56 TQFP64 SDIP56 TQFP64	None None None None	None None None None	ST72T671N6B6	5 samples sent with this spec

Notes:

x:ROM size code

y: Temperature code (6 or 3)

z:Reset option (blank or S)

(1):By default, STMicroelectronics provides only samples of the representative devices mentioned as available.

Upon specific request and in case the product is not terminated, STMicroelectronics can provide any other device

(2):Product termination in progress

(3):Only OTP sales types are mentioned

(4):Ceramic package are used for EPROM devices

REQUEST FOR ST7 PROGRAMMING ALGORITHM

Fax to STMicroelectronics at (33) 04 76 58 56 26
or mail to STMicroelectronics
Avenue des Martyrs
#BP 217
38019 GRENOBLE CEDEX FRANCE

Attn: Development Tools - F. AUCLAIR

YES, I wish to automatically receive the new version of ST7 Programming Algorithm when it is released

Please send me 5 samples of ST7 devices in addition to devices already received.

ST7 Representative Device Name: . . . . . .

(See the list of existing devices in Appendix of ST72EXX JTAG PROGRAMMING SPECIFICATIONS)

USER REGISTRATION DATA

Name: . . . . . .

Company: . . . . . .

Address: . . . . . .

. . . . . .

. . . . . .

City: . . . . . .State:. . . . . .

ZIP code: . . . . . .Country: . . . . . .

Telephone number: . . . . . .

Fax number: . . . . . .

"THE PRESENT NOTE WHICH IS FOR GUIDANCE ONLY AIMS AT PROVIDING CUSTOMERS WITH INFORMATION REGARDING THEIR PRODUCTS IN ORDER FOR THEM TO SAVE TIME. AS A RESULT, STMICROELECTRONICS SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WITH RESPECT TO ANY CLAIMS ARISING FROM THE CONTENT OF SUCH A NOTE AND/OR THE USE MADE BY CUSTOMERS OF THE INFORMATION CONTAINED HEREIN IN CONNEXION WITH THEIR PRODUCTS."