Features

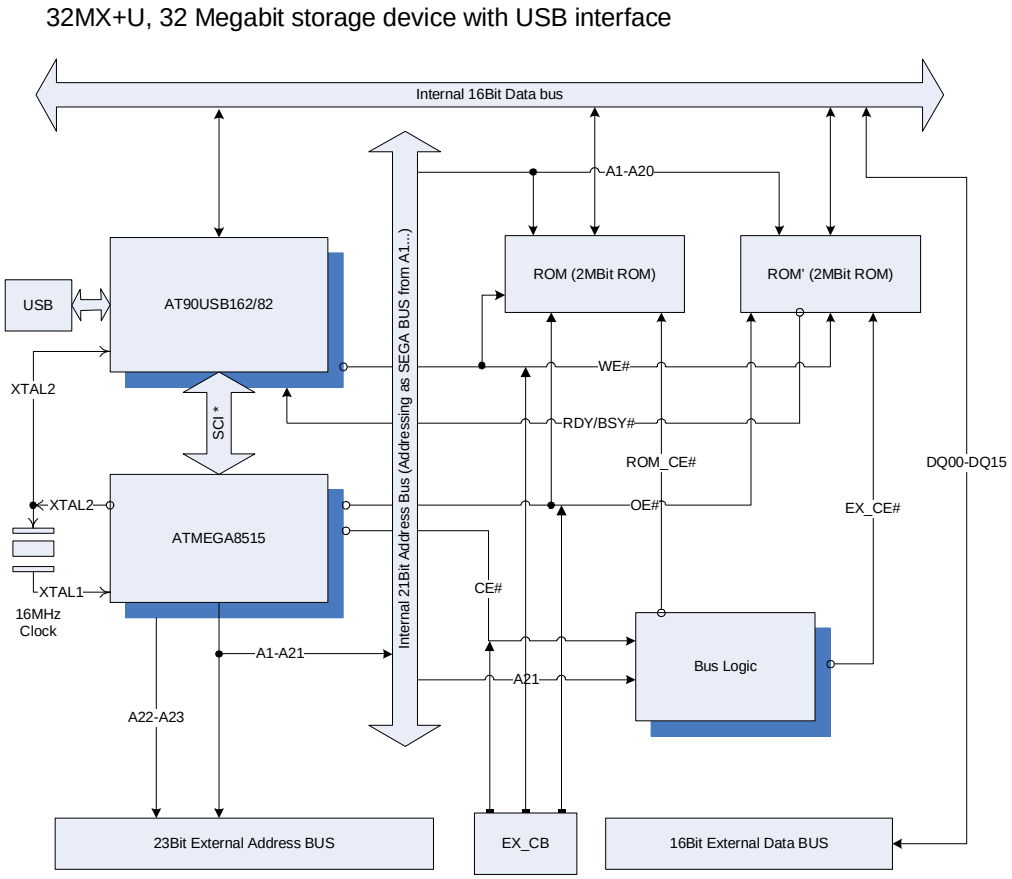
• USB 2.0 Full-speed Device Module with Interrupt on Transfer Completion Controller
  – Complies fully with Universal Serial Bus Specification REV 2.0
• Firmware upgrades support (USB using)
• ISP onboard programming interface (separated for USB/Data controller and Address controller)
• Nonvolatile 32 Megabit onboard flash memory
  Endurance: 1,000,000 Write/Erase Cycles.
• SCI implementation for master/slave operations.
• Peripheral Features:
  – 23 Bit external Address BUS.
  – 16 Bit external Data BUS.
  – 3 external IO interface pins.
• Operating Voltages
  – 4.6 - 5.5V for USB wire flashing.
  – 4.5 - 5.5V for other external operations.
• Speed Grades
  – 50c maximum Full memory erasing time (nominal 23c, minimum 7c).
  – 82.2 KByte/c guarantied speed for SCI Operations.
  – 1.1 MBit/c minimum speed for USB Operations.
  – 57.2 MBit/c guarantied speed for external operations with onboard memory.

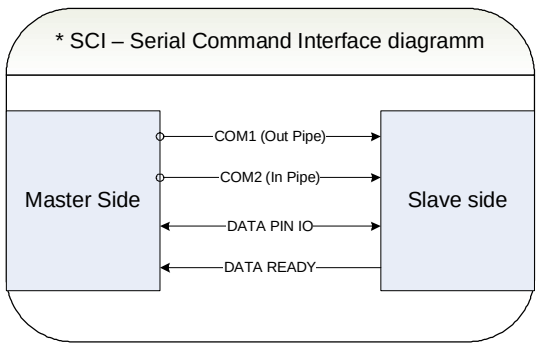
Overview

The 32MX+U is a storage device based on the AVR CMOS 8-bit microcontrollers (AT90USB162/82 and ATMEGA8515, enhanced RISC architecture) with USB data storage processing.

Block Diagram

Figure 1. Block Diagram.





The 32MX+U core combines a data and address BUSes. All pins may be reached on external lamel connector.

Internal data (AT90USB) and address (ATMEGA8515) controllers connected with SCI interface, data controller connected as Master and directly load commands into address controller from USB host.

Data controller also configured as USB device (can't be connected as USB HOST).

USB interface connection

32MX+U Device has mini USB connector. It directly connects USB wires and AT90USB AVR device. While 32MX+U device connected to USB wire it can't be processed any external (non USB) operations.

Data BUS

Data BUS is 16Bit bus, drive by data controller. All data bits can be programmed as input, or output. Data range from 0x0000 to 0xFFFF.

Address BUS

Internal and external addresses BUSes directly connected to address controller (ATMEGA8515). External BUS can addressed by controller from 0x000000 to 0x7FFFFF, while internal address BUS using only address range from 0x000000 to 0x1FFFFF, A21 pin directly connected to logic unit, it means that FLASH ROMs addressed only by 19 addresses lines (A1-A20) and bank selection operations processed by BUS Logic Unit (importance CE# and A21 pins).

BUS Logic Unit

BUS logic unit make driving flash ROMs banks. BUS logic, very simple logic device, please see this Truth **Table 1**.

Table 1. BUS logic truth table.

CE#	A21	ROM_CE#	EX_CE#	Selected Bank
0	0	0	1	ROM
0	1	1	0	ROM'
1	0	1	1	None
1	1	1	0	ROM'

Board Clock

Board Clock source use Atmega8515 internal RC Oscillator, stabilized 16MHz crystal, Oscillator produce clock for Atmega8515 CPU Core, Output from the Oscillator amplifier directly connected to AT90USB internal clock operating circuit input. This method makes working AT90 and ATMEGA CPU Cores synchronously.

SCI Pin description Master/Slave Side

All input pins on maser side must be configured in HIGH-Z state, on slave size must be pull upped.

DATA PIN IO Pin for data input/output.

DATAREADY Slave state report pin. High – slave Ready, otherwise – Busy. Input pin on Master, output on Slave.

COM1 Command for data input. Maser send data bit to slave.
Then master change on this pin logic level from HIGH to LOW (falling edge activate) slave must start read input data bit from DATA PIN IO. Then data bit is read, slave report it with HIGH level on DATAREADY pin.

COM2 Command for data output. Maser read data bit from slave.
Then master change on this pin logic level from HIGH to LOW (falling edge activate) slave must set next data bit onto DATA PIN IO. Then data bit is set, slave report it with HIGH level on DATAREADY pin.

SCI

Serial Command Interface

SCI interface connects address and data controllers it produces several simple command (see **table 2**), every command consists **Command header**, **Command size** (optional) and **Command Data Field** (if data size presented), summary command packet can't be longer then 128 bytes (1byte Command Header + 1byte Command size + 126bytes Command data filed, illustrated on **figure 2**), (command can be sent only from Master side):

Recommended packet size is 62 bytes.

Figure 2. SCI Packet simple.

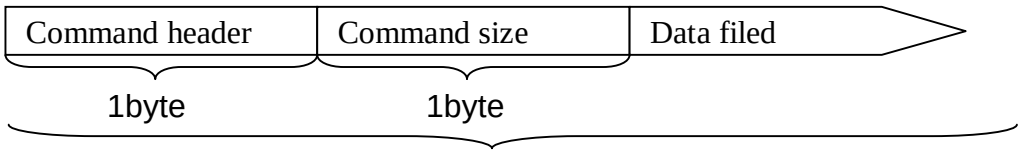
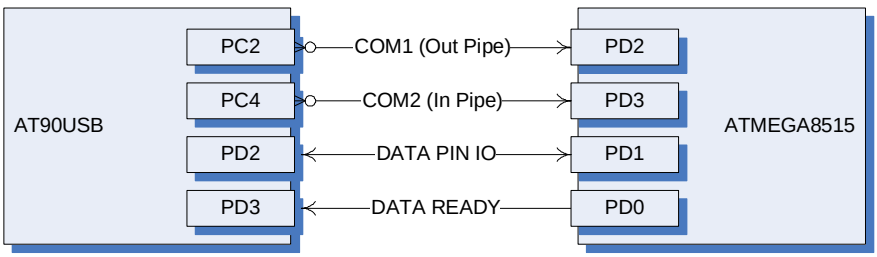


Table 2. SCI Commands.

Command header	Mark	Size (byte)	Value	Description
INIT_8515	S	1	0x53	Start using 8515
EXIT_8515	U	1	0x55	Release using 8515
RESET_8515	Q	1	0x41	Perform reset device
EX_BUF	E	>1	0x45	Send ex buffer
INC_A_W	W	>1	0x57	Send start address for writing
INC_A_R	R	>1	0x52	Send start address for reading
ROM_IN_ST	I	1	0x49	Set ROM In state
ROM_OUT_ST	O	1	0x4F	Set ROM Out state
RAM_IN_ST	A	1	0x41	Ram functions
BOOT_LOAD	B	1	0x42	Init firmware upgrade process

SCI Interface implementation

Figure 3. AT90USB and Atmega8515 SCI Connection diagram:

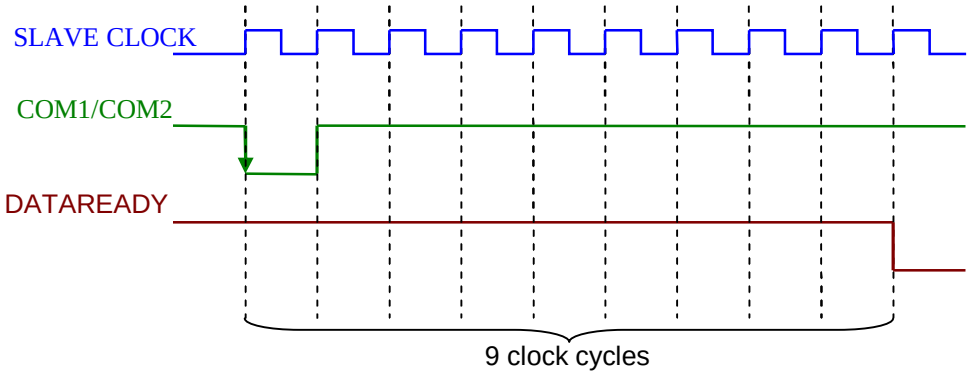


SCI, AT90USB connected as master, Atmega8515 connected as slave.

Slave controlled by master on COM1, COM2 interrupts pins. If DATAREADY line holds by slave device in HI State, it understands by master, - "slave can receive or send data to/from".

Then Master device clock COM1/COM2 pin, slave must process corresponding function in 9 clock cycles period and report it as busy state (DATAREADY – LO State) **diagram 1**.

Diagram 1. Slave device reaction time.

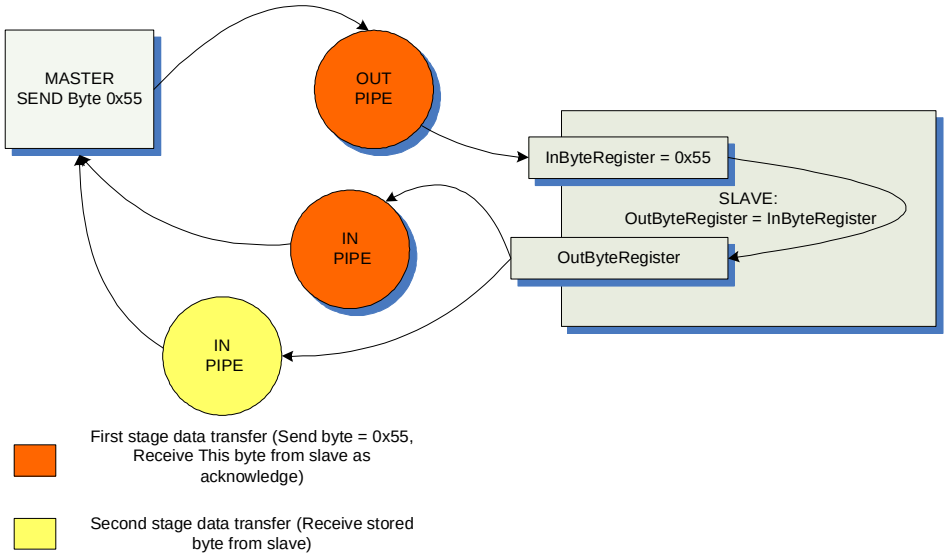


SCI Data transfer and acknowledgements

SCI algorithm for slave module does not include transfer error check. All transfer errors checking operations realized on master side.

Then master sends any byte data to slave, slave store received byte to output buffer, this data byte can be read back by master from slave IN PIPE. To understand this please see **figure 4**.

Figure 4. Read/Write by Master Module sample.



Master side error checking

Master module understands error then transmitted and read back transmitted byte not equal. To understand how can master module correct error lets see command processing in slave module.

Commands processing by slave

Then slave receive **1 byte command**, go to command state, it means, slave wait for IN PIPE to process received command and for send acknowledge byte.

Then receive **multi byte command**, slave goes to preprocess command state, wait for command **size** by master from OUT PIPE and IN PIPE to send acknowledge byte.

Then receive command size, slave wait for **Command Data Field**. Then command data field received, slave process start command condition and wait for activation on COM2 pin (falling edge). Falling edge on COM1 will return slave to IDLE.

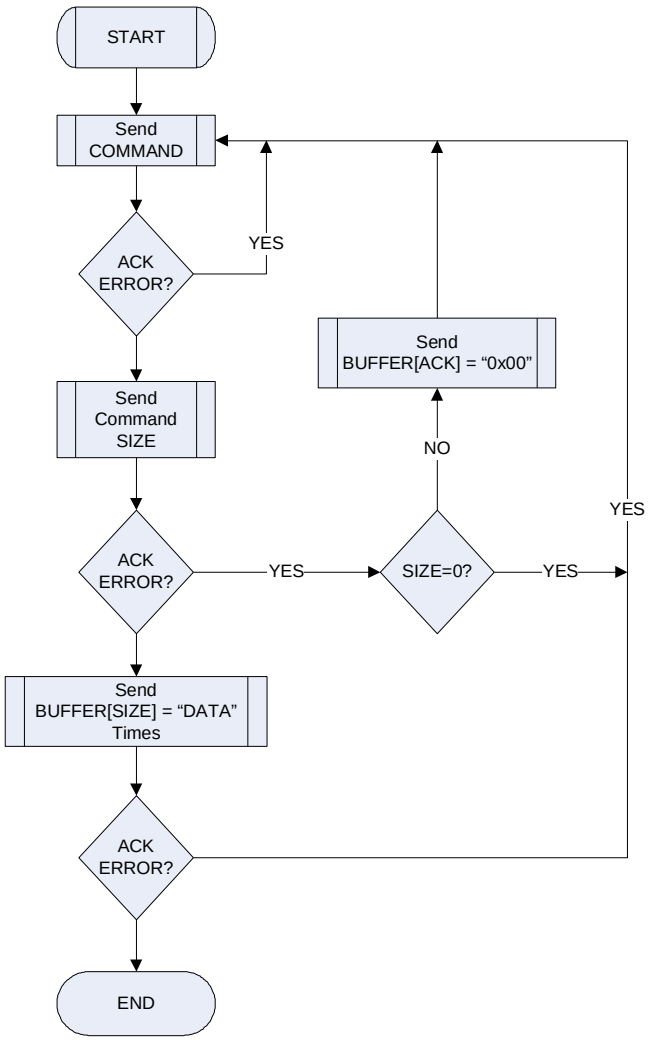
Master side error handling

Master receives error then sends to and received from slave bytes not equal. It means that master must run **Repair command process**.

Repair command process

Repair command process run by master then error while transmit received. See **figure 5**.

Figure 5. Send command algorithm and Repair command process.



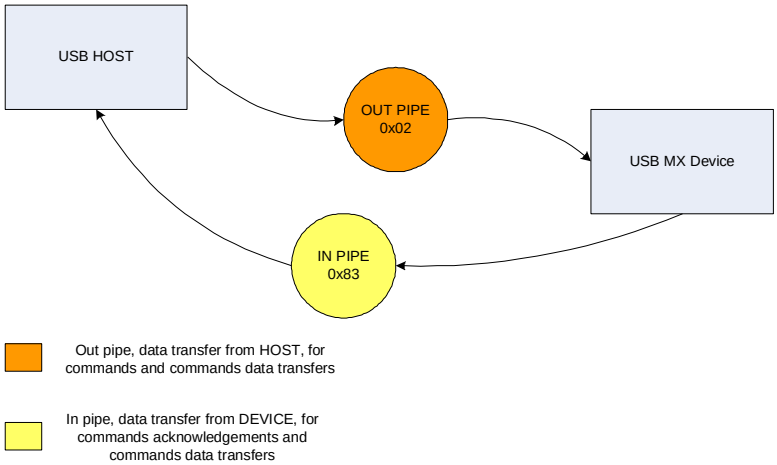
! Important SCI note

Data transfer from master to slave, must consists 8 stages because slave process any command function only then full byte received, and so, while master start out pipe process, Slave clear **in pipe bit** counter and counter can’t be reached 8 bit (1 byte received by slave).

USB
communication with ROM

USB communication with ROM controlled by software. Software control MX series device by sending command to USB Pipes 0x02 and 0x83.

Figure 6. USB communication overview.



Summary MX USB command format

The USB command size is 31 byte. It consists 3 parts, **Constant Header** filed, **Command** filed, **Command parameters** field.

Table 3. Command packet, MX USB device

Position in bytes	Name	Size in bytes	Value	Description
0-3	**sConstantHeader**	4	“USBC”	Standard SCSI command header
4-15	**lpbReserved**	11	0x00	Reserved field
16	**bMySCSICommand**	1	0x67	My extended SCSI command
17	**bMXCommand**	1	See table 4	MX commands field
18-30	**lpbCommandParams**	14	See MX Commands definitions	Command Parameters

MX USB commands

MX USB device controller must know how to process twelve main commands listed in **table 4**.

Table 4. Valid MX USB device Commands.

Name	Value	Description
ATM_READY	0x22	Perform ADDR BUS firmware update process
SEC_DEVID	‘L'	Read installed ROM chip device ID
SEC_RAM_WRITE	'U'	Perform write 256 bytes to RAM
SEC_RAM_READ	'D'	Perform read 256 bytes from RAM
SEC_READY	'C'	Report ROM Status
SEC_ERASE	‘E'	Perform erase sector
SEC_WRITE	'W'	Set start write address and write 64Bytes-16KBytes sector
SEC_READ	‘R'	Set start read address and read 256bytes-16KBytes sector
SEC_PUT_NAME	'P'	Perform write file name
SEC_GET_NAME	'G'	Perform read file name
SEC_BOOTLOAD	‘B'	Init boot load process
SEC_ADDR_BUST	'K'	Test address BUS

Firmware Upgrade

Processes Overview

Firmware Upgrade Processes consists two stages, first stage, - Upgrade Address BUS firmware, second stage, - Upgrade USB/DATA firmware. Upgrade USB/DATA firmware can't be making by "it self", device can't write New firmware while Old firmware working, So, then upgrade process initializing USB/DATA controller detach "32MX+U" device from USB HOST, and attach upgrade firmware "DFU BOOTLOADER". When all update processes completed, USB/DATA controller performs reconnection to new firmware.

! Important firmwares upgrade note

While any Upgrade stages perform, do not detach 32MX+U device from USB wire and do not close any upgrade software, it will take device permanently unworkable!!!