The block diagram of the processor

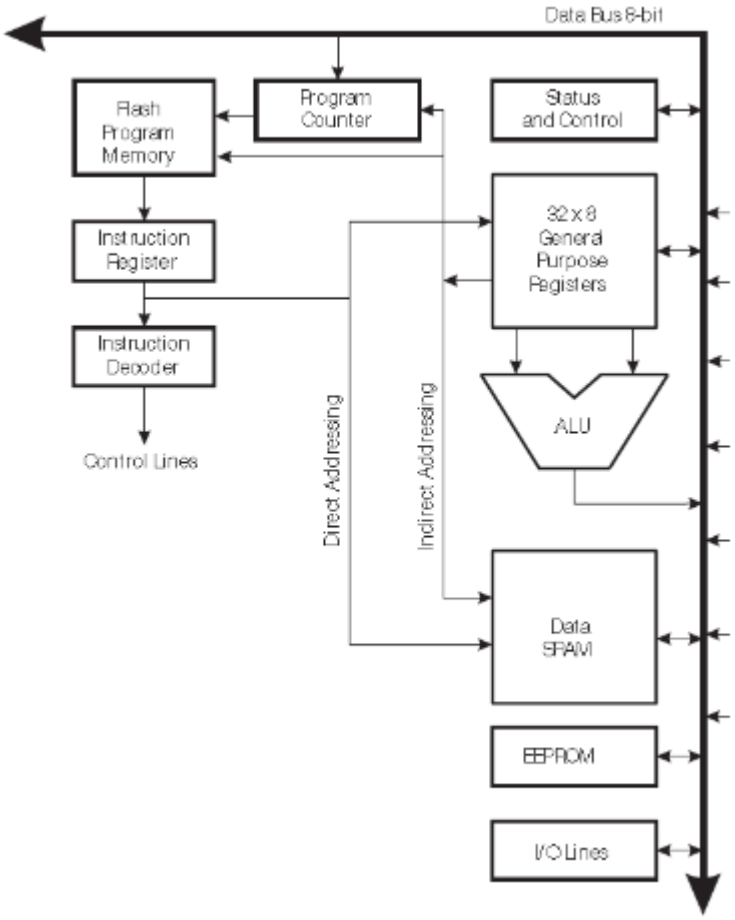

Figure 1. AVR Architecture

CPU Architecture inside. Continuous line, where most of the blocks will be connected, which is called Data Bus 8-bits. The next important block call the Program Memory, it is the type of memory, which is going to restore the programs executing this architecture – Flash memory. The size of this member changes from module to module generally, but we have in our architecture 256KBytes. It is connected to the Data Bus and can exchange data with the rest of modules. After that from this Program memory what we are doing is to obtain the instructions to be executed by the rest of the architecture. Once we get instructions, instruction is in store – in a special register, Instruction Register. It also has a size of memory to load the data, which comes from the Flash memory and then the microcontroller has to decode this instruction. The decoded information will be located in another block Instruction Decoder, which will provide numerous control signals that will be distributed all along the architecture controlling different blocks.

Another important module is Program Counter or register, which is also connected to the bus, but the most important that it is connected to the Program memory. The Counter block contain the address of the next instruction that will be executed by the microcontroller. The number in

this register is used as an address to go to the program memory and read the instruction, which will be loaded in the Instruction register and then decoded.

Most of the instructions are going to a temporary store – Register Files. It composed of 32 registers, in each one of them has 8 bits. There is a place where most of the temporary results of the instructions executed by the microcontroller are stored. This block has a connection to the 8-bit Data Bus, but no less important connections is between Registers File and Program Memory. That means that if we want to get some data from the register file and use it as an address to access the program memory. Another important functionality from the register file is to change a value of a Register Counter.

From the Register File there are additional outputs, to the ALU (Arithmetic Logic Unit). This block is ready to receive to operates, one of them comes from the Register File. By the way, the Register file gets directly the address from the Instruction Register. ALU is performing arithmetical, logical or mathematical operations and writing its values on the data bus.

The Data Memory of type SRAM is implemented to that the program manipulates certain data at its store in this chip. It has a connection to the Data Bus and one connection from the Register File, and connection from the Instruction Register to control the block. From the Register File we can get the data, which can be written in the SRAM memory.

Status Register is in charge of remembering certain events that occur while instructions are executing in the data path. All the instructions conditions are placed here in 8-bit register and they are stored as a 0 or 1 condition.

The AVR microcontroller also specialized in I/O operations, it has plenty of the modules that are in charge a performing different I/O operations in all of them are individually connected to the bus. All of them can be controlled also with machine instructions executed within the CPU.

Timers block performs certain tasks related to timers, which is also connected to the bus. ADC comparator, performs a logical task with compare signals. And the last one – Interruptions of microcontroller, it includes interrupts, which can be called within the CPU (Global interrupts, Reset interrupts, Timers interrupts and etc).

Instruction set

Instructions are the machine commands in machine language, which are obtaining in the CPU by the Program memory (Flash) where the instructions can be executed by the rest of the architecture. Once we get instructions, instruction is in store – in a special register, Instruction Register. It also has a size of memory to load the data, which comes from the Flash memory and then the microcontroller has to decode this instruction. The decoded information will be located in another block Instruction Decoder, which will provide numerous control signals that will be distributed all along the architecture controlling different blocks. The Program Counter or register block, which is also connected to the bus in the CPU, but the most important that it is connected to the Program memory. The Counter block contain the address of the next instruction that will be executed by the microcontroller.

The list of all instructions we can find on page 416 of the doc2549 manual about Flash Microcontroller programming.

Power management

When we have an application for embedded system without external power supply, thus energy efficient is very important. The AVR microcontrollers have a several special modes to manage power – Sleep Modes. These modes allows the application to shut down unused modules in the MCU or shut down modules soon after using. For example, Analog-Digital Conversion can be switched off after measurement.

SM2	SM1	SM0	Sleep Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby
1	1	1	Extended Standby

Figure 2. Sleep Mode Select

Sleep modes switch of clocks (CPU clock, Flash memory clock and etc), but stay awake some sources, such as USART, ADC and others. Each mode has its own influence specifications for different parts of microcontroller. If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, executes the interrupt routine, and resumes execution from the instruction following SLEEP.

Idle Mode.

When MCU is in Idle mode by instruction SLEEP, this mode stops the CPU, but allowing he SPI, USART, Analog Comparator, ADC, Timer/Counters and the interrupt system works. It halts only CPUclk and FLASHclk, while allowing the other clocks to run. MCU wakes up from external triggered interrupts as well as internal (Timer Overflow and USART transmit complete) interrupts. To reduce consumption in the Idle mode, the Analog Comparator can be powered down, if it doent required for application.

ADC Noise Reduction.

When MCU is in this mode by instruction SLEEP, this mode stops the CPU, but allows the ADC, external interrupts, Timer/Counter2 and Watchdot operating (if enebled). It halts CPUclk and FLASHclk, while others allowed to run. This improves the noise environment for the ADC, it enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically for this mode. MCU can wake up from the ADC Converion Complete interrupt, an External Reset, a Watchdog System Reset or Interrupt, Timer/Counter2 interrupt and SPM/EEPROM ready interrupt.

Power-down Mode.

In this Sleep Mode the external Oscillator is stopped, while interrupts and the 2-wire Serial Interface and the Watchdog operating (if enabled by user). Only External Reset, a Watchdog

Reset, the 2-wire Serial Interface address match, an external level interrupts or a pin change interrupt can wake up the MCU. The MCU can be woken up by a level triggered interrupt, but it will require some time to wake up after the changed level. There is a delay from the wake-up condition occurs until the wake-up becomes effective.

Power-Save Mode.

The same as a Power-down Mode, except:

If Timer/Counter2 is enabled, running happens during sleep. The device can wake up from the Timer Overflow or Output Compare event, if interrupt enable bits on TIMSK2, and the Global Interrupt Enable bit in SREG. If Timer/Counter2 is shut down, the recommended mode is Power-down Mode.

Standby Mode.

This mode is identical to Power-down with the exception that the Oscillator is kept running. The device wakes up in six clock cycles in this Standly mode.

Extended Standby Mode.

This mode is identical to Power-save mode with the exception that the Oscillator is kept running. The device also wakes up in six clock cycles as in Standby mode.

Memory System

The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition EEPROM Memory for data storage. There are three in sum. Program Memory, it is the type of memory, which is going to restore the programs executing this architecture – Flash memory. The size of this member changes from module to module generally, but we have in our architecture 256KBytes. It is connected to the Data Bus in the CPU and can exchange data with the rest of modules.

The Data Memory of type SRAM is implemented to that the program manipulates certain data at its store in this chip. It has a connection to the Data Bus and one connection from the Register File, which is a place where most of the temporary results of the instructions executed by the microcontroller are stored. That means that if we want to get some data from the register file and use it as an address to access the program memory. From the Register File we can get the data, which can be written in the SRAM memory. There can be also I/O, Extended I/O addresses.

Atmega2560 pins

I/O ports such as PORTx (where small letter “x” is A, B, C, D, E, F, G, H, J, K or L) are 8-bit bi-directional I/O ports with internal pull-up resistors (selected for each bit). Each of these ports serves the functions of special features of the Atmega2560. Each of them has their special propose, which make them unique: PORTB has better driving capabilities than others ports, or PORTF serves as analog inputs to the A/D Converter. It also serves the functions of the JTAG interface. If the JTAG interface is enables, the pull-up resistors on pins PF7 (TDI), PF5 (TMS) and PF4 (TCK) will be activated even if a reset occurs.

PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	READ PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WRx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE	WPx:	WRITE PINx
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE	$clk_{I/O}$:	I/O CLOCK
SLEEP:	SLEEP CONTROL	DIxn:	DIGITAL INPUT PIN n ON PORTx
PTOExn:	Pxn, PORT TOGGLE OVERRIDE ENABLE	AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Figure 3. Alternate Port Functions

Because of most of the ports look the same, we can discuss about some of the most interesting PORTs for us, explained below, because of their feature for programming. All the pins call the Change Pin interrupt and pins can serve as an external interrupt source, for example to wake up the MCU from the Sleep Mode. Each of them has Timer/Counterx or Output Compare from Px7-Px4, then for example in PORTD there is the most interesting alternative function, answering for interrupts of the USART Transceiver and Receiver, see below:

PORTD

NT3/TXD1 – Port D, Bit 3. INT3, External Interrupt source 3: The PD3 pin can serve as an external interrupt source to the MCU. TXD1, Transmit Data (Data output pin for the USART1). When the USART1 Transmitter is enabled, this pin is configured as an output regardless of the value of DDD3.

INT2/RXD1 – Port D, Bit 2. INT2, External Interrupt source 2. The PD2 pin can serve as an External Interrupt source to the MCU. RXD1, Receive Data (Data input pin for the USART1). When the USART1 receiver is enabled this pin is configured as an input regardless of the value of DDD2. When the USART forces this pin to be an input, the pull-up can still be controlled by the PORTD2 bit.

PORTE

AIN1/OC3A – Port E, Bit 3. AIN1 – Analog Comparator Negative input. This pin is directly connected to the negative input of the Analog Comparator.

AIN0/XCK0 – Port E, Bit 2. AIN0 – Analog Comparator Positive input. This pin is directly connected to the positive input of the Analog Comparator.

Negative and Positive voltages in A/D converter of the AVR used for differential conversion.

PORT F

This port has an alternate function as analog input for the ADC (PF0-PF3) and enabling JTAG debugging interfaces on pins (PF4-PF7).

Port Pin	Alternate Function
PF7	ADC7/TDI (ADC input channel 7 or JTAG Test Data Input)
PF6	ADC6/TDO (ADC input channel 6 or JTAG Test Data Output)
PF5	ADC5/TMS (ADC input channel 5 or JTAG Test Mode Select)
PF4	ADC4/TCK (ADC input channel 4 or JTAG Test ClocK)
PF3	ADC3 (ADC input channel 3)
PF2	ADC2 (ADC input channel 2)
PF1	ADC1 (ADC input channel 1)
PF0	ADC0 (ADC input channel 0)

Figure 4. Port F pins Alternate Functions

PORT K

Some ports like Port K are used for Analog-to-Digital Conversations (PK0-PK7) occurs (ADC8-ADC15). Others ports looks the same between each other, the difference between them, that they have another numeric number of a Timer/Counter/External Interrupt (from 0-5) on pins 0,1, 4, 6, 7.

Example: PORT D

Port Pin	Alternate Function
PD7	T0 (Timer/Counter0 Clock Input)
PD6	T1 (Timer/Counter1 Clock Input)
PD5	XCK1 (USART1 External Clock Input/Output)
PD4	ICP1 (Timer/Counter1 Input Capture Trigger)
PD3	$\overline{\text{INT3}}$/TXD1 (External Interrupt3 Input or USART1 Transmit Pin)
PD2	$\overline{\text{INT2}}$/RXD1 (External Interrupt2 Input or USART1 Receive Pin)
PD1	$\overline{\text{INT1}}$/SDA (External Interrupt1 Input or TWI Serial DAta)
PD0	$\overline{\text{INT0}}$/SCL (External Interrupt0 Input or TWI Serial CLock)

Figure 5. Port D pins Alternate Function

T0 – Port D, Bit 7. T0, Timer/Counter0 counter source.

T1 – Port D, Bit 6. T1, Timer/Counter1 counter source.

XCK1 – Port D, Bit 5. XCK1, USART1 External clock. The Data Direction Register (DDD5) controls whether the clock is output (DDD5 set) or input (DDD5 cleared). The XCK1 pin is active only when the USART1 operates in Synchronous mode.

ICP1 – Port D, Bit 4. ICP1 – Input Capture Pin 1: The PD4 pin can act as an input capture pin or Timer/Counter1.

INT0/SCL – Port D, Bit 0. INT0, External Interrupt source 0. The PD0 pin can serve as an external interrupt source to the MCU.

How to set signals for Alternate Functions to the PORTx pins see (8-bit Microcontroller with 256K Bytes In-System Programmable Flash Atmega2560) manual p.78-95.

VCC – is the digital supply voltage for processor.

AVCC – is the supply voltage pin for PORTF and the A/D Converter, which is externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

AREF – is the analog reference pin for the A/D Converter. Can be configured in the application for different types of ADC conversions.

GND – Ground.

XTAL1 and XTAL2 are the input and output the inverting Oscillator amplifier. XTAL1 also is input to the internal clock operating circuit.

RESET – is the reset input.