Technological University of the Philippines
Manila Campus
College of Industrial Technology
Electronics Engineering Technology

Name: _____
Course/Year/Section: _____
Date Started: _____ *Date Submitted:* _____
Assessed by: _____ *Rating:* _____

MICROCONTROLLER OVERVIEW

Module 2

I. OBJECTIVES

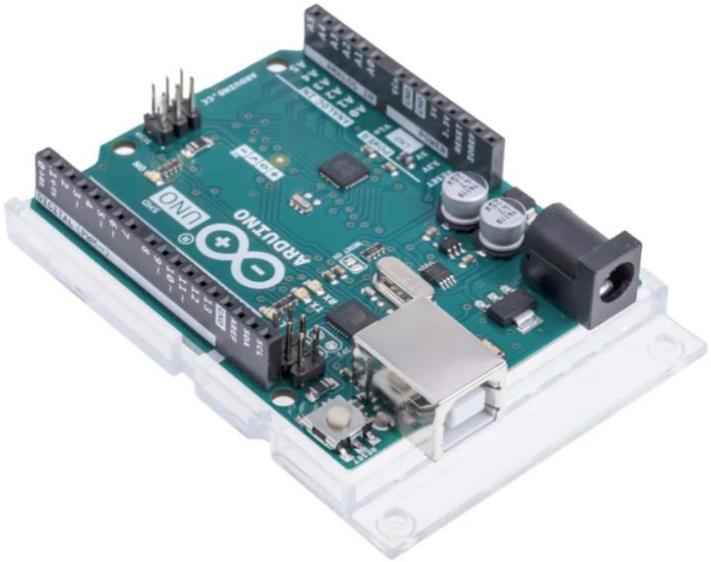
At the end of the lesson, the students are expected to:

1. Understand the fundamentals of a microcontroller unit, i.e., Arduino Uno and Atmel ATMega328p.
2. Grasp the technical significance of the microchip microcontroller as an intelligent digital switching device.
3. Design a block diagram that showcases the different parts/components/modules/registers/other system peripherals and map out the program and data memories including their address locations

II. GENERAL INFORMATION

1. **ARDUINO UNO –** a microcontroller unit (MCU) based on the ATmega328P processor (Atmel/ Microchip Technology, 2016) developed by Arduino.cc, and originally introduced in 2010. “Uno” means “one” in Italian which marks the major redesign of an open-source platform that is compatible with multiple hardware and software resources. It has 14 digital input/outputs, 6 of which can be used as Pulse Width Modulation (PWM) outputs; 6 analog inputs, a 16-20MHz ceramic resonator (clock speed), a USB connection, a power jack, an ICSP header and a reset button.

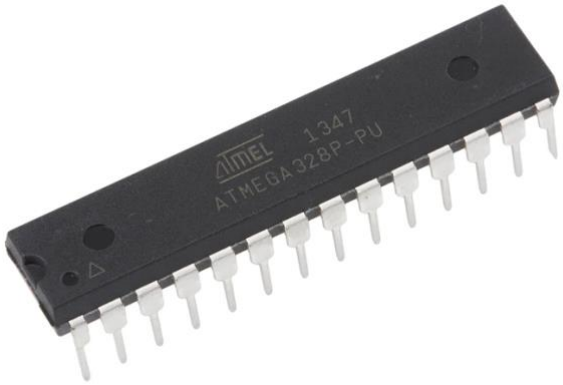
Figure 1
Arduino Uno R3



a. ATmega328P Processor

- 8-Bit, Alf and Vegard’s Reduced Instruction Set Computer (RISC) CPU
- 32KB Flash (large, non-volatile program memory with storage capacity)
- 2KB SRAM (Static Random-Access Memory used for high-speed cache memory, etc.)
- 1KB EEPROM (Electrically Erasable Programmable Read-Only Memory, used for storing small amount of data even when power is off)
- 131 powerful instructions mostly under single clock cycle execution
- 32x8 general purpose registers (GPR)
- 10,000 flash/ 100,000 EEPROM write/erase cycles

Figure 2
Atmel ATMega328P



b. Security

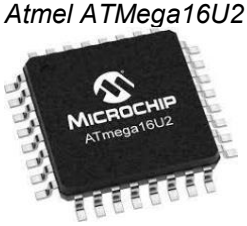
- Power On Reset (POR) – resets the MCU at predefined state (e.g., ideal voltage value) upon initiation.
- Brown Out Detection (BOD) – prevents the MCU in operating in unstable voltage threshold.

c. Peripherals

- 2x 8-bit; and 1x16-bit Timer (times precise intervals) /Counter (counts the number of events) with a dedicated period register and compare channels for Pulse Width Modulation (PWM).
- 1x Universal Synchronous/Asynchronous Receiver/Transmitter (USART, used for low-speed local, serial communication) with fractional baud rate generator and start-of-frame detection
- 1x controller/peripheral Serial Peripheral Interface (SPI, used for high-speed full duplex communication with sensors, displays, etc. in short distances)
- 1x Dual mode controller/peripheral Inter-Integrated Circuit (I2C, a two-wire bus used for half duplex communication with sensors, displays, etc.
- 1x Analog Comparator (AC) used to compare two analog voltage inputs and generate comparative results
- Watchdog Timer with separate on-chip oscillator, which serves as a monitoring system and resets the MCU in case of program bug.
- Six PWM channels, used to control analog-like signals, such as those used in sensors,
  LEDs, motor speeds, servo position etc.
- Interrupt and wake-up, which allows the MCU to enter in a low-power sleep mode and wake up when a specific pin changes

d. ATMega16U2 Processor – used in managing the Universal Serial Bus (USB) to serial communication with the main processor; it is a 8-bit AVR RISC. It has 16 KB In-System Programming Flash (ISP, used for bootloader installation, firmware updates etc.); 512B EEPROM; 512B SRAM debugWIRE, an interface for on-chip debugging and programming

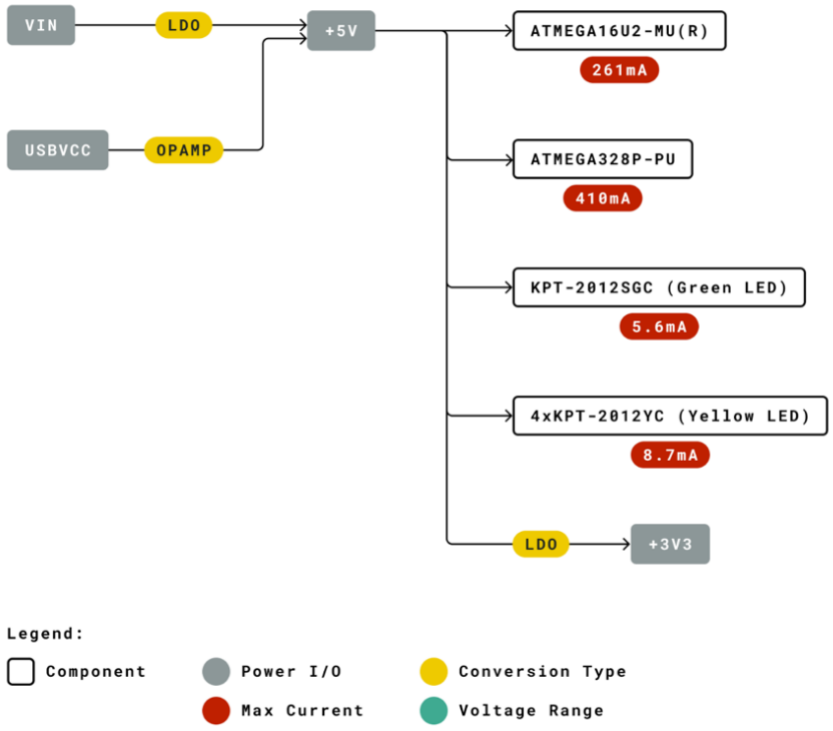
Figure 3
Atmel ATMega16U2 Processor



e. Power – it operates from 2.7-5.5 volts with the following power tree:

- VIN – voltage in (can range from 7-12V then converted to 5V) from an external power source
- USBVCC – voltage supplied by a USB communication protocol
- LDO – low dropout regulator; provides stable and regulated voltage output.
- OPAMP – operational amplifier used to stabilize analog signals
- Operational temperature: -40 to +125°C

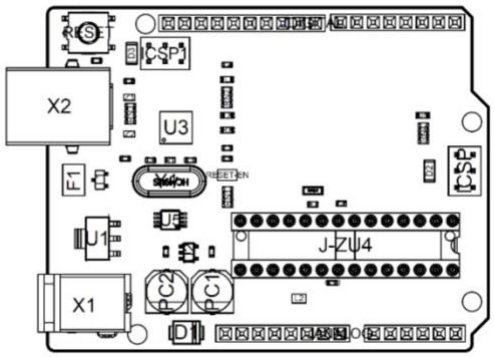
Figure 4
Arduino Uno Power Tree Diagram



2. PIN CONFIGURATION

a. Board Topology

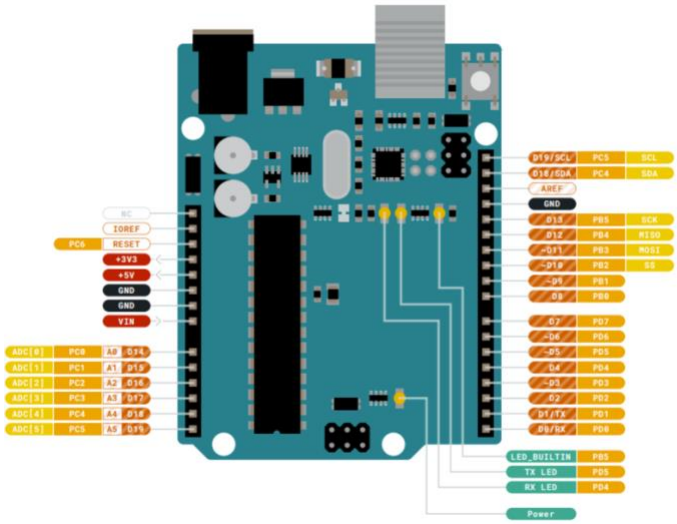
Figure 5
Arduino Uno Board Typology



Ref.	Description	Ref.	Description
X1	Power jack 2.1x5.5mm	U1	SPX1117M3-L-5 Regulator
X2	USB B Connector	U3	ATMEGA16U2 Module
PC1	EEE-1EA470WP 25V SMD Capacitor	U5	LMV358LIST-A.9 IC
PC2	EEE-1EA470WP 25V SMD Capacitor	F1	Chip Capacitor, High Density
D1	CGRA4007-G Rectifier	ICSP	Pin header connector (through hole 6)
J-ZU4	ATMEGA328P Module	ICSP1	Pin header connector (through hole 6)
Y1	ECS-160-20-4X-DU Oscillator		

b. Connector Pinout

Figure 6
Arduino Uno Pinouts



b.1 Analog Pinout

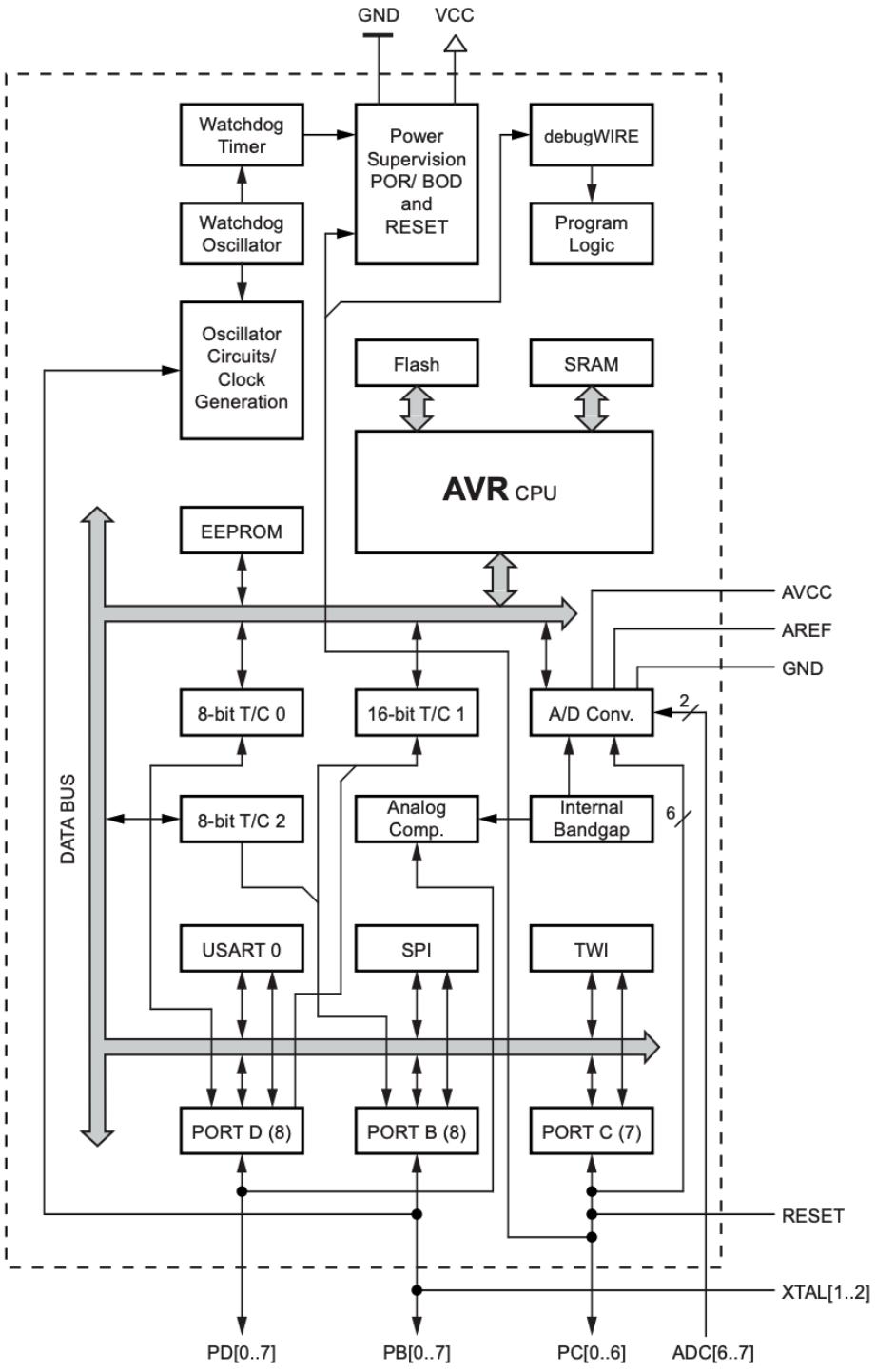
Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

b.2 Digital Pinout

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

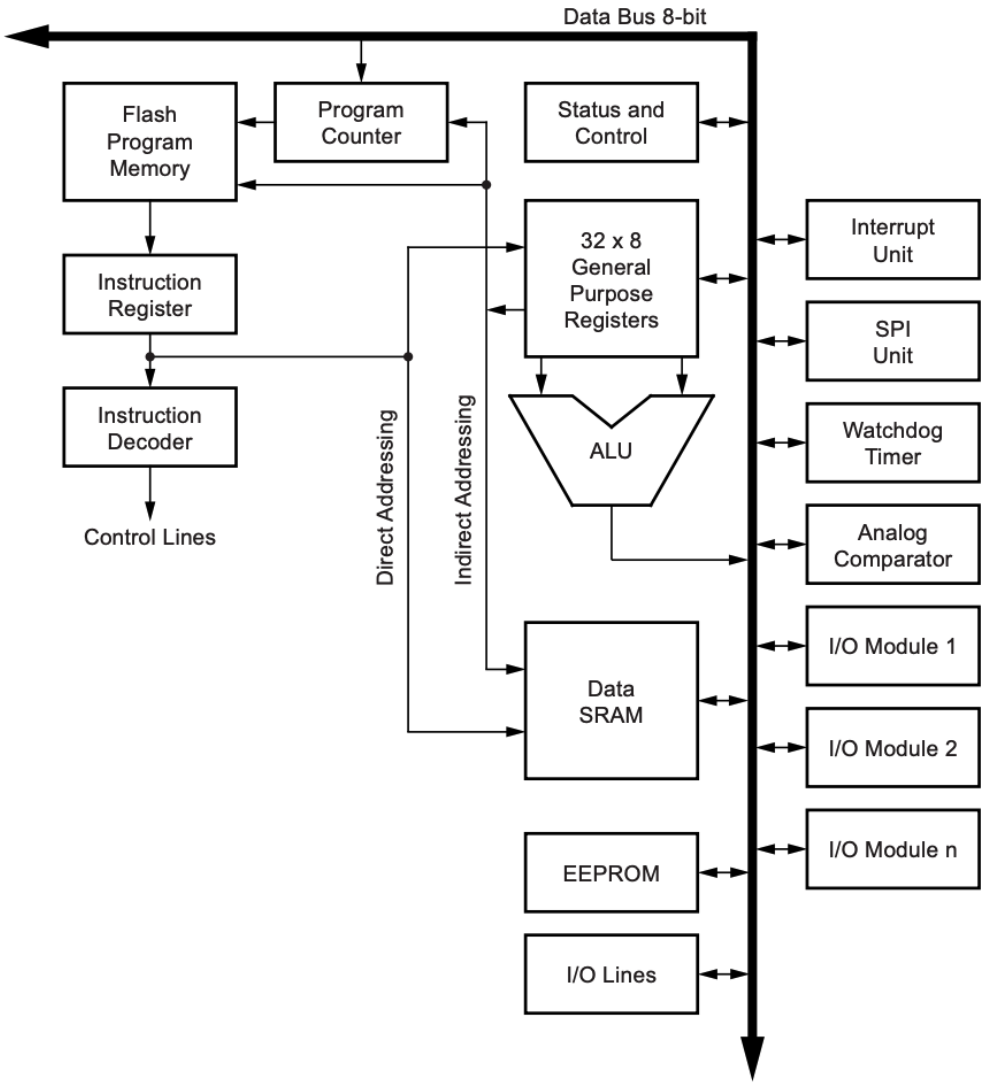
3. ATMEGA 328P BLOCK DIAGRAM

Figure 7
Atmel ATMega328P Block Diagram



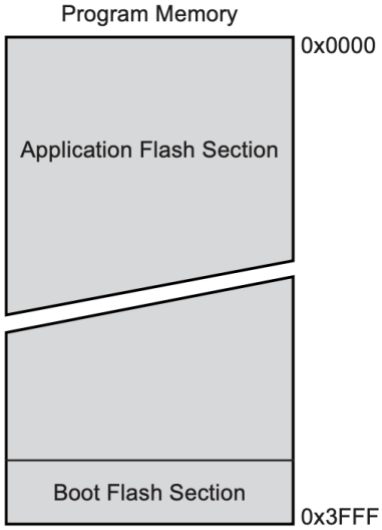
4. AVR CPU (HARVARD) ARCHITECTURE

Figure 8
Atmel ATMega328P Architecture



a. **Flash Program Memory –** type of a non-volatile memory used to store the program code; during execution, the MCU fetches and runs instruction from this memory (read-only) max of 32KB. It is divided in two sections, the boot program section and the application program section. Both sections have dedicated lock bits for write and read/write protection. The Store Program Memory/EEPROM instruction that writes into the application flash memory section must reside in the boot program section. The flash memory has an endurance of at least 10,000 write/erase cycles.

Figure 9
Atmel ATMega328P Program Memory Map



b. **Instruction Register** – a register used in the execution of instructions, which works hand-on-hand with a program counter; it fetches instructions from flash memory in sequential and temporary order. The AVR has 131 powerful instruction which work with single clock cycle execution *(See page 281-284 for Instructions samples).*

c. Instruction Decoder – identifies and decodes instructions to be processed by the ALU, such as arithmetic or logical operations, in coordination with the GPR and Data SRAM.

d. Program Counter – also known as instruction pointer works in conjunction with the Instruction Register. It monitors and updates the memory address where the next instruction will be fetched. The ATmega328P program counter (PC) is 14 bits wide, thus addressing the 16K program memory locations.

e. **Arithmetic Logic Unit (ALU) -** operates in direct connection with all the 32 GPRS. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic (addition, subtraction, increment, decrement), logical (AND, OR, XOR, NOT), and bit manipulation functions (set, clear, toggle, shift left, shift right).

f. Status Register – a special purpose register which represents and flags the outcome of various operations and instructions; it contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the status register is updated after all ALU operations, as specified in the instruction set reference. *See Page 11 for more details*

Figure 10

Atmel ATMega328P Status Register Map (SREG)

Bit	7	6	5	4	3	2	1	0	
0x3F (0x5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

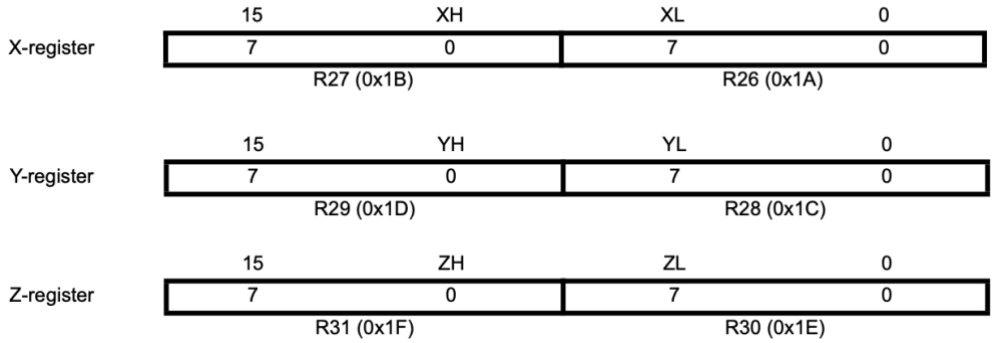
g. **General Purpose Register (GPR) –** versatile memory used for a wide range of data manipulation purposes; it stores data, addresses, or intermediate results during program execution. The AVR has 32 GPR, 12 of which are directly connected to the ALU, which allows 2 independent registered to be accessed in a single instruction cycle per clock cycle. Most of the instructions operating on the register file have direct access to all registers, and most of them are single cycle instructions. Each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user data space. *See pages12-13 for more details.*

Figure 11
Atmel ATMega328P GPR Map

	7 0	Addr.	
	R0	0x00	
	R1	0x01	
	R2	0x02	
	…		
	R13	0x0D	
General	R14	0x0E	
Purpose	R15	0x0F	
Working	R16	0x10	
Registers	R17	0x11	
	…		
	R26	0x1A	X-register Low Byte
	R27	0x1B	X-register High Byte
	R28	0x1C	Y-register Low Byte
	R29	0x1D	Y-register High Byte
	R30	0x1E	Z-register Low Byte
	R31	0x1F	Z-register High Byte

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the data space. The three indirect address registers are the **X** (R27:26, index or pointer for data memory), **Y** (R29-28, index or pointer for data memory; indirect hums and calls in assembly language) and **Z** (R31-30, reading instructions for program/flash memory, accessing I/O registers and EEPROM memory, and other special function registers etc.).

Figure 12
X, Y, and Z GPRs



h. Data SRAM – a volatile memory used as temporary storage of data during program execution. It is separate from the program (flash memory) and serves specific purposes such as variable storage for arrays, and other data structures, stack memory for function calls and returns, alternate GPR when there is an overspill, data buffering for USART, SPI, and I2C, data management for tracking purposes of various I/O devices, and real-time data, as well as arithmetic and logical operations. The SRAM has 2KB storage capacity and the **internal data SRAM access takes 2 clock cycles** (clk_{CPU}).

The following figure shows the data memory map. The first 32 x 8-bit locations address the register file (GPR, special, R0-R31 or 0x0000-0x001F), the next 64x8-bit addresses the standard I/O memory (0x0020-0x005F), then another 160x8-bit locations for extended I/O memory (0x0060-0x00FF), and the last 2048x8-bit locations address the internal data SRAM (0x0100-0x08FF).

Figure 13
Data SRAM Map

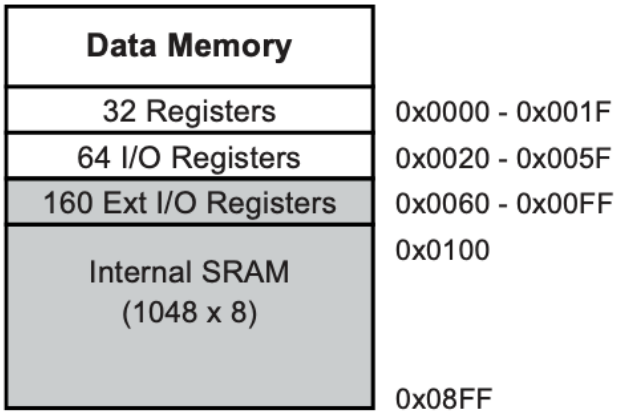
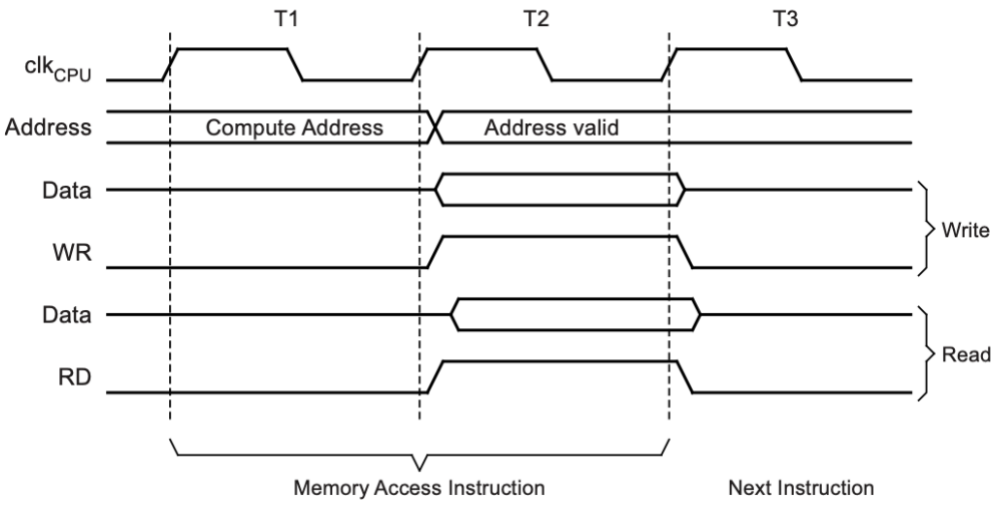


Figure 14
On-Chip Data SRAM Access Cycles



i. **Stack Pointer (SP)** – stores temporary data, e.g. local variables, and return addresses after interrupts and subroutine calls. It always points to the top of the stack. The PUSH command decreases the stack pointer. The SP is implemented as 2 8-bit registers in the I/O space. The AVR stack pointer is implemented as two 8-bit registers in the I/O space. *See page 13-14 for more details.*

j. **EEPROM –** Electrically Erasable Programmable Read-Only Memory is a non-volatile memory used for storing small amount of data (1KB max) amidst power shutdown or reset; for configuration parameters; calibration and reading of sensors; counter and cumulative totals; security-related data; firmware updates; and bootloader configurations etc. The EEPROM has an endurance of at least 100,000 write/erase cycles. Note: When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed *See pages 20-23 for the EEPROM-related registers*

k. **Watchdog Timer -** has a separate on-chip oscillator, which serves as a monitoring system and resets the MCU in case of program bug.

l. **Oscillator Circuit / clock generation system** – used to generate clock for the execution timing of instructions in the MCU and is supplied various clock systems. The device is shipped with internal RC oscillator at 8.0MHz and with the fuse CKDIV8 programmed, resulting in 1.0MHz system clock. *See pages 24-33 for Clock-related registers*

Figure 15
Clock Systems

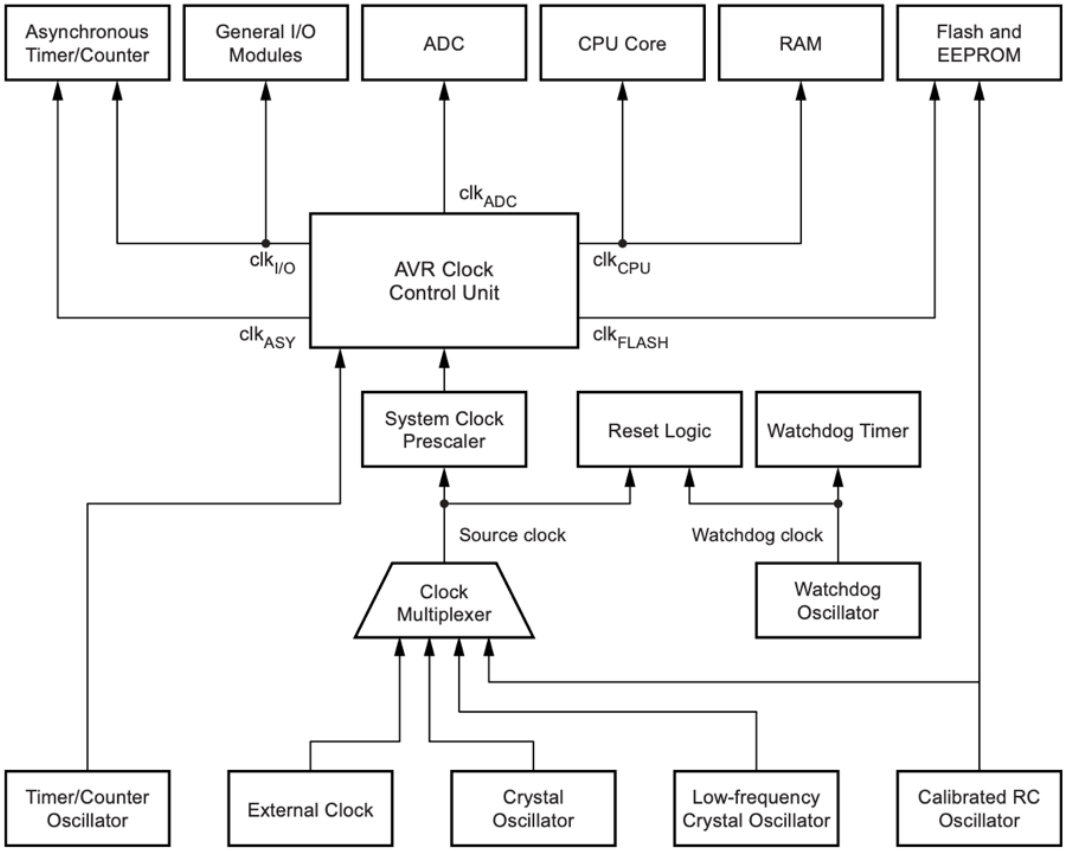


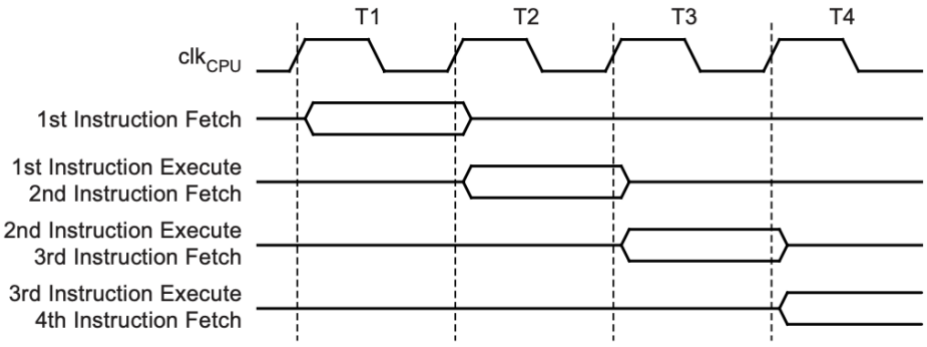
Table 1
Device Clocking Options

Device Clocking Option	CKSEL3..0
Low power crystal oscillator	1111 - 1000
Full swing crystal oscillator	0111 - 0110
Low frequency crystal oscillator	0101 - 0100
Internal 128kHz RC oscillator	0011
Calibrated internal RC oscillator	0010
External clock	0000
Reserved	0001

Note: 1. For all fuses "1" means unprogrammed while "0" means programmed.

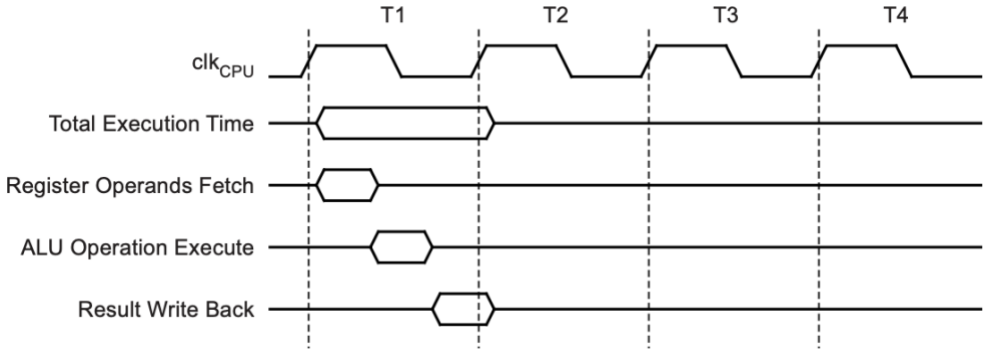
The following figure shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 16
8-Bit AVR Instruction Execution Timing



Specifically, in a single clock cycle, the ALU uses 2 register operands *(e.g., ADD = Rd (destination register), Rr (source register and added to destination register)*

Figure 17
Single Cycle ALU Operation



m. **Serial Peripheral Interface (SPI)** - used for synchronous high-speed full duplex data communication with sensors, displays, etc. in short distances)
n. **Interrupt unit** – also known as wake-up unit, which allows the MCU to enter in a low-power sleep mode and wake up when a specific pin changes
o. **Analog Comparator (AC)** - used to compare two analog voltage inputs and generate comparative results
p. **Status and Control Registers** (SFIOR) - used to manage and monitor various microcontroller operations. Status register is used to provide flags or information on the overall status of the MCU. Control registers are used for general control, watchdog timer, power reduction, sleep control, timer/counter control, and I/O control, AC control, and communication interface control.

III. CHECKPOINT

As group, work on the following requirements:

1. In reference your module 1 requirement (i.e., IPO model), design a block diagram (with flowcharts) that showcases the different parts/components/modules/registers/other system peripherals that elaborate the relationship of all inputs and outputs. Think beyond the box. (25 points)

2. Assuming you are using Atmel ATMega328P MCU for the IPO model process module, map out the program and data memories including their address locations that you will use to characterize the overall operation of the system. Specify the architecture to use. (25 points)

3. Specify the percentage contribution of each group member (100% total). Each member should sign to confirm the breakdown of contribution.