ATmega328P

Narendiran S

July 28, 2020

article
[a4paper, bottom=0.5in, top=0.5in, left=0.5in, right=0.5in]geometry wrapfig natbib url xcolor caption hyperref byte-field
amsfonts float enumitem
graphicx
ATmega328P Basics Narendiran S July 28, 2020

0.1 Features

- 8 bit CMOS μC with RISC Architecture
- 32 x 8-bit General purpose registers
- 32 KByte of flash program memory
- 1 KByte EEPROM
- 2 KByte of internal SRAM
- On-chip 2-cycle multiplier
- Optional boot code section with independent lock bits
  - In-system programming by on-chip boot program
  - True read-while-write operation
- Two 8-bit Timer/Counter with separate prescaler and compare mode
- One 16-bit Timer/Counter with separate prescaler, compare mode and capture mode
- Real time counter with separate oscillator
- Six PWM channels
- 6/8(DEPENDING ON PACKAGE) chaneel 10 bit ADC ∘ Also with Temperature measurement
- Programmable serial USART
- 2-wire serial interface (Phillips I2C compatible)
- Programmable watchdog timer with separate on-chip oscillator
- On-chip analog comparator
- Interrupt and wake-up on pin change
- Power-on reset and programmable brown-out detection
- External and internal interrupt sources
- Six sleep modes: Idle, ADC noise reduction, power-save, power-down, standby and external standby
- 2.7V to 5.5V for ATmega328P

0.2 Block Diagram

blockDiagram.png

0.3 Pins

0.3.1 Power Pins

VCC, Gnd - 2.7V to 5.5V

0.3.2 PORTB - PB7:PB0

- Bidirection I/O with internal pull-up resistor(selectable for each bit)
- Tristate when reset
- Depending on the clock selection fuse settings,
  - *PB6* – input of inverting oscillator amplifier and input to internal clock operating circuit
  - *PB7* – output of inverting oscillator amplifier
- If internal calibrated RC oscillator is used as clock source, *PB7* and *PB6* is used as *TOSC2* and *TOSC1* input for Timer/Counter2

0.3.3 PORTC - PC5:PB0

- Bidirection I/O with internal pull-up resistor(selectable for each bit)
- Tristate when reset

0.3.4 PC6/$\overline{RESET}$

- Low level on this pin will gnerate reset, even if no clock running.
- ***RSTDIBL*** fuse == programmed(0) – *PC6* is input pin.
- ***RSTDIBL*** fuse == unprogrammed(1) – *PC6* is reset pin.

0.3.5 PORTD - PD7:PD0

- Bidirection I/O with internal pull-up resistor(selectable for each bit)
- Tristate when reset

0.3.6 AV_{CC}

- Supply voltage pin for A/D converter
- Connected to External Vcc when not used
- Connected to Vcc through LPF when used

0.3.7 AREF

Analog reference pin of A/D Converter

0.3.8 ADC7:ADC6

Analog input to ADC(10bit ADC)

0.4 Modes

0.4.1 Idle Mode

Stops the CPU while allowing SRAM, TImer/Counters, USART, 2-wire serial interface, SPI port and interrupt system to continue functioning.

0.4.2 Power-Down Mode

Saves the register contents but freezes the oscillator, disabling all other chip functions untill next interrupt or hardware reset.

0.4.3 Power-Save Mode

The asynchonous timer continues to run, allowing user to maintain timer base while reset of devices is sleeping.

0.4.4 ADC Noise reduction mode

Stops CPU and all I/O modules except asynchonous timer and ADC to minimize switching noise during ADC conversions.

0.4.5 Standby Mode

The crystall/oscillator is running while reset of devices is sleeping. Allows very fast start-up combined with low power consumption.

0.5 AVR CPU Core

The main function of CPU core is access memory, perform caluclations, control peripherals and handle interrupts.

avrCoreBlockDiagram.png

- For performance and parallelism, the AVR uses Harvard Architecture - with seperate memories and buses for program and data.
- Instructions in Program memory are exectued with a single level pipelining.
- The program memory is **In-system Reprogrammable Flash memory**.
- The register file consist of 32 x 8-bit General Purpose Registers with a single clock cycle assess time.
- One ALU operation uses two operatands from register file and store back the result to register file in one clock cycle.
- Six 32-bit register combine to form the X-, Y- and Z- registers which help in 16-bit indirect address register pointer for data space.
- One of these pointers acts as address pointer for look-up tables in Flash Program Memory.
- Program memory adress cotains 16-bit or 32-bit Instructions.
- Program Flash memory space is divided into two sections - each section have dedicated lock bits for read/write protection.
  - Boot Program section
  - Application Program section
- I/O memory space contains 64 addresses for CPU peripheral functions as control register, SPI and Other I/O functionscan be accessed directly or through register file from 0x20 - 0x5F.
- Has extended I/O space from 0x60 - 0xFF in SRAM.

0.5.1 Reset and Interrupt vectors

- Interrupts and reset vectors have seperate program vecotr in program memory space.
- Interrupts maye be disbaled when boot lock bits ***BLB02*** or ***BLB12*** are programmed.
- Lowest ddresses in program memory space are reset and interrupt vectors.
- The lower the addess the higher the priority.
- RESET has the highest followed by INT0(the external interrupt request 0).
- The interrupt vectors can be moved to start of boot flash section by setting ***IVSEL*** bit of **MCUCR** (MCU control register).
- THe reset can be moved to start of boot flash sectio by programming the ***BOOTRST*** fuse.

0.5.2 Interrupt Handling

- The ***I-bit*** (global interrupt enable bit) of **Status register** must be enabled.
- When a interrupt occurs, ***I-bit*** (global interrupt enable bit) is cleared and all interrupts are disabled.
- The user can write logic one to ***I-bit*** to enable nested interrupts.
- The ***I-bit*** is automatically set when returning from interrupt Instructions.

0.6 AVR Memories

Two main memory spaces - Data memory and Program memory space and a EEPROM memory for data storage.

0.6.1 In-System Reprogrammable Flash Program Memory

- 32 KBytes on-chip in-system reprogrammable flash memory for program space.
- Since, the Instructions are all 16-bit or 32-bit wide, the flash(program space) is organized as 16K x 16.
- Endurance of atleast 10,000 write/erase cycle.
- For software security, Flash program memory space is divided into
  - Boot Loaded section
  - Application Program section
- The Program Counter is bits wide and thus can address 16K program memory location.

`programMemoryFlash.png`

0.6.2 SRAM Data Memory

- The ATmega328P is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the opcode for the IN and OUT instructions.
- For the extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

`dataMemorySRAM.png`

- The lower 2303(0x08FF) data memory locations addresses both the register files, the I/O memory, extended I/O memory and the internal data SRAM.
  - The first 32 location addresses the register file.
  - The next 64 location addresses the standard I/O memory.
  - The following 160 location address the extended I"O memory.
  - The last 2048 location address the internal data SRAM.

0.6.3 EEPROM Data Memory

- 1 K Byte of data EEPROM memory.
- Organized as seperate data space.
- Endurance of atleast 100,000 write/erase cycle.
- EEPROM are accessible in I/O space.
- Specific Write procedure is followed.

0.6.4 I/O Memory

- I/O and peripherals are placed in the I/O spaces.
- All I/O locations are accesed by LD/LDS/LDD and ST/STS/STD instructions.
- The I/O registers withing 0x00 - 0x1F are directly bit-accessible using SBI and CBI Instructions.

0.7 System Clock and Clock Options

`clkDistribution.png`

0.7.1 Clock Systems

CPU Clock

- clk_{CPU} is routed to all parts of AVR core.
- General purpose register file, Status register and data memory holding stack pointer.
- Halting CPU clock will inhibts the core from perfrorming general operations and caluclations.

I/O Clock

- $clk_{I/O}$ is used in I/O modules like Timers/Counter, SPI, USART, etc.
- For external interrupt module also but some external interrupts are detected by asynchonous logic and can be used even when I/O clock is halted.

Flash Clock

- clk_{FLASH} controls operation of flash interface.

Asynchronous Timer Clock

- clk_{ASY} allows asynchonous Timer/Counter to be clocked directly from external clock or an external 32 kHz clock crystall.
- This clock allows using Timer/COunter as real-time counter even when device is in sleep mode.

ADC Clock

- $clk_{ADC} had dedicated clock domain$
- Gives more accurate ADC conversion result

0.7.2 Clock Sources

Selectable clock sources using flash fuse bits.

CKSEL[3:0]	**Device Clocking Option**
1111 - 1000	Low power crystall oscillator
0111 - 0110	Full swing crystal oscillator
0101 - 0100	Low frequency crystal oscillator
0011	Internal 128kHz RC oscillator
0010	Calibrated internal RC oscillator
0000	External clock

For fuses, ”1” denotes unprogrammed and ”0” denotes programmed.

Default Clock Source

- Devices is shipped with interface RC oscillator at 8.0MHz with fuse ***CKDIV8*** programmed meaning $----->$ the internal oscillator produces a 8.0 Mhz clock but due to ***CKDIV8*** being programmed the system clock gets $\frac{8.0MHz}{8} = 1MHz$.
- The startup time is set to maximum and time-out period enabled.
- Default configuration $--->$ ***CKSEL*** = 0010; ***SUT*** = 10; ***CKDIV8*** = 0.

Clock Start Sequence

- Clock Source needs a sufficient V_{CC} and minimum number of oscillating cycles before stablizing.
- To ensure sufficient V_{CC}, the device issues an internal reset with time-out delay (t_{TOUT}).
- The number of cyles in the dealy is set by ***SUTx*** bits and ***CKSELx*** fuse bits.
- The main purpose of dealy is to keep AVR in reset until it is supplied with minimal V_{CC}.
- The start-up sequence for the clock includes both the time-out delay and the start-up time when the device starts up from reset.

Clock Output Buffer

- device can output system clock on the *CLKO* pin.
- enabled by *CKOUT* fuse.
- any clock source can be used to output from this pin.

TIMER/COUNTER OSCILLATOR

- uses the same crystal oscillator for low-frequency oscillator and Timer/Counter oscillator.

- Since, It shares the Timer/Counter oscillator pins *TOSC1* and *TOSC2* pins with *XTAL1* and *XTAL2*, the system clock must be four times the oscillator and so the Timer/Counter oscillator can only be used when the calibrated internal RC oscillator is selected as system clock source.

Low Power Crystall OSscillator

`lowPowerCrystallOscillatorCircuit.png`

Figure 1: VGA Connector

CKSEL[3:1]	**Frequency Range (MHz)**
100	0.4 to 0.9(only Ceramic resonators)
101	0.9 to 3.0
110	3.0 to 8.0
111	8.0 to 16.0

- *XTAL1* and *XTAL2* are inputs and outpus of an inverting amplifier which can be configured as on-chip oscillator.

- Either Quartz Crystall or Ceramic resonator can be used.

- Crystal Oscillator is a low power oscillator with reduced voltage swing on the XTAL2 output.

- Not capable of driving other clock inputs.

- C1 and C2 should be of the same values – 12pF to 22pF.

- The ***CKSEL[0]*** fuse together with the ***SUT[1:0]*** fuses select the start-up times as shown in Table below.

startUpTimesLowPowerCrystalOscillator.png

Full Swing Crystal Oscillator

fullSwingCrysallOscillatorCircuit.png

Figure 2: VGA Connector

CKSEL[3:1]	**Frequency Range (MHz)**
011	0.4 to 16.0

- *XTAL1* and *XTAL2* are inputs and outpus of an inverting amplifier which can be configured as on-chip oscillator.
- Either Quartz Crystall or Ceramic resonator can be used.
- Full-Swing with rail-to-rail swing on the XTAL2 outtput.
- Can drive other clock input

- Power consumption is more than Low power crystal oscillator
- Needs V_{CC} = 2.7 to 5.5V
- C1 and C2 should be of the same values – 12pF to 22pF.
- The ***CKSEL[0]*** fuse together with the ***SUT[1:0]*** fuses select the start-up times as shown in Table below.

startUpTimesLowPowerCrystallOscillator.png

Low Frequency Crystal Oscillator

- To use with 32.765kHz watch crystal
- Crystal Cap(CL) – 6.5,9.0 and 12.5pF
- ***CKSEL[3:0]*** == 0101.
- The Start-up Times for the Low-frequency Crystal O scillator Clock Selection

startUpTimelowFrequencyCrysalOscillator.png

Calibrated Internal RC Oscillator

- 8.0MHz clock
- Voltage and temperature dependent
- Calibration is done in **OSCCAL**.
- Default mode shipeed with CKDIV8 prescalar programmed to prescale causing the system clock to be 1.0MHz.
- ***CKSEL[3:0]*** == 0010.

startUpTimeCalibratedInternalRCOscillator.png

128kHz Internal Oscillator

- low power oscillator with 128kHz frequency

- ***CKSEL[3:0]*** == 0010.

startUpTimeCalibratedInternalRCOscillator.png

External Clock

externalClockciruit.png

- XTAL1 must be connected to external source.
- 0 - 16 MHz frequency.
- ***CKSEL*** == 0000.

- Start-up times are determined by the SUT fuses as

startUpTimeExternalClock.png

0.7.3 System Clock Prescalar

- The system clock can be divided by setting the **CLKPR** (Clock Prescale Registers) value.
- Used to decrease the system clock frequency and the power consumption when the requirement for processing power is low.
- Affects the clk_{SYS}, clk_{IO}, clk_{ADC}, clk_{CPU} and clk_{FLASH}.
- A special write procedure is followed to change ***CLKPS*** bits:

  (i) Write the clock prescaler change enable (***CLKPCE***) bit to one and all other bits in **CLKPR** register to zero.

  (ii) Within four cycles, write the desired value to ***CLKPS*** bit while writing a zero to ***CLKPCE***.

  (iii) Interrupt must be disabled.

Register Description

OSCCAL – Oscillator Calibration Register

7	6	5	4	3	2	1	0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0

- The oscillator calibration register is used to trim the calibrated internal RC oscillator to remove process variations from the oscillator frequency.
- A pre-programmed calibration value is automatically written to this register during chip reset.
- The application software can write this register to change the oscillator frequency.
- If EEPROM is to be used, shouldn't do calibration for more than 8.8 MHz.
- ***CAL7*** bit detected range of operation of oscillator. Setting zeros gives the Lowest requency range, setting this bit to 1 gives the highest frequency range.
- The ***CAL[6:0]*** bits are used to tune the frequency within the selected range. A setting of 0x00 gives the lowest frequency in that range, and a setting of 0x7F gives the highest frequency in the range.

CLKPR – Clock Prescale Register

7	6	5	4	3	2	1	0
CLKPCE	-	-	-	CLKPS3	CLKPS2	CLKPS1	CLKPS0

- ***CLKPCE*** - Cloc k Prescaler Change Enable must be written to logic one to enable change of the ***CLKPS*** bits.
- The ***CLKPCE*** bit is only updated when the other bits in CLKPR are simultaneously written to zero.

CLKPS[3:0]	**Clock Division Facter**
0000	1
0001	2
0010	4
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256

- ***CLKPS[3:0]*** - Clock Prescaler Select Bits define the division factor between the selected clock source and the internal system clock.
- The ***CKDIV8*** fuse determines the initial value of the ***CLKPS*** bits.
  - If ***CKDIV8*** is unprogrammed, the ***CLKPS*** bits will be reset to 0000.
  - If ***CKDIV8*** is programmed, ***CLKPS*** bits are reset to "0011", giving a division factor of 8 at start up.
- Note that any value can be written to the ***CCLKPS*** bits regardless of the ***CCKDIV8*** fuse setting.

0.8 Power Management and Sleep modes

- Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power.
- When enabled, the brown-out detector (BOD) actively monitors the power supply voltage during the sleep periods.
- To further save power, it is possible to disable the BOD in some sleep modes.

0.8.1 Sleep Modes

- To enter any of the six sleep modes, the ***SE*** bit in **SMCR** register must be written to logic one.
- The ***SM[2:0]*** bits in the **SMCR** register select which sleep mode.
- *SLEEP* instruction must be executed.
- 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.
- The contents of the register file and SRAM are unaltered when the device wakes up from sleep.
- If a reset occurs during sleep mode, the MCU wakes up and executes from the reset vector.
- The Active Clock Domains and Wake-up Sources in the Different Sleep Modes,

sleepModeDomain.png

Idle Mode

- Stops the CPU but allows the SPI, USART, analog comparator, ADC, 2-wire serial interface, Timer/Counters, watchdog, and the interrupt system.
- Halts clk_{CPU} and clk_{FLASH} and allows other clocks.
- Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the timer overflow and USART transmit complete interrupts.

ADC Noise Reduction Mode

- Stops the CPU but allows ADC, the external interrupts, the 2-wire serial interface address watch, Timer/Counter2 and the watchdog.
- Halts $clk_{I/O}$, clk_{CPU} and clk_{FLASH} and allows other clocks.
- Improves the noise environment for ADC, enabling higher resolution measurement.
- ADC Noise Reduction Mode enables the MCU to wake up from external reset, a watchdog system reset, a watchdog interrupt, a brown-out reset, a 2-wire serial interface address match, a Timer/Counter2 interrupt, an SPM/EEPROM ready interrupt, an external level interrupt on INT0 or INT1 or a pin change interrupt.

Power-down Mode

- Stops the external oscillator but allows the external interrupts, the 2-wire serial interface address watch, and the watchdog.
- Halts all clocks and asynchronous modules only.
- Power-down mode enables the MCU to wake up from an external reset, a watchdog system reset, a watchdog interrupt, a brown-out reset, a 2-wire serial interface address match, an external level interrupt on INT0 or INT1, or a pin change interrupt.

Power-save Mode

- Only diffence from Power-down mode is Timer/Counter2 is enabled and it will run.
- Timer overflow or output compare event from Timer/Counter2 can wake up.

Standby Mode

- Selects the external crystal clock option.
- Identical to power-down except oscillator is running.

External Standby Mode

- Selects the external crystal clock option.
- Identical to power-Save except oscillator is running.

Register Description

SMCR – Sleep Mode Control Register

7	6	5	4	3	2	1	0
-	-	-	-	SM2	SM1	SM0	SE

SM[2:0]	Sleep Mode
000	Idle
001	ADC Noise Reduction
010	Power-down
011	Power-save
110	Standby
111	External Standby

- ***SE*** bit must be written to logic one just before the SLEEP instruction is executed, to make the MCU enter the sleep mode.

0.8.2 Power Reduction Register

- To stop the clock to individual peripherals to reduce power consumption.
- The current state of the peripheral is frozen and the I/O registers can not be read or written.
- Peripheral should in most cases be disabled before stopping the clock.
- Wake up peripherals can be done by writing zero to bits in **PRR**.

Register Description

PRR – Power Reduction Register

7	6	5	4	3	2	1	0
PRTWI	PRTIM2	PRTIM0	-	PRTIM1	PRSPI	PRUSART0	PRADC

Bits	Name
PRTWI	Power Reduction TWI
PRTIM2	Power Reduction Timer/Counter2
PRTIM1	Power Reduction Timer/Counter1
PRTIM0	Power Reduction Timer/Counter0
PRSPI	Power Reduction Serial Peripheral Interface
PRUSART0	Power Reduction USART0
PRADC	Power Reduction ADC

0.8.3 Minimizing Power Consumption

- In general, sleep modes should be used as much as possible.
- ADC should be disabled before entering any sleep mode.
- Analog comparator should be disabled in all sleep modes.
- If the brown-out detector is not needed by the application, this module should be turned off by ***BODLEVEL*** fuses.
- If Internal Voltage Reference is not needded and ADC or analog comparator or BOD is not needed, the Internal Voltage Reference can be disabled.
- If the watchdog timer is not needed in the application, the module should be turned off.
- If On-chip Debug System is not needed, can be disabled by ***DWEN*** fuse.
- For Port pins,
  - No pins drive resistive loads.
  - Input buffers are disabled when I/O clock and ADC clocks are stopped.
  - If the input buffer is enabled and the input signal is left floating or have an analog signal level close to V CC /2, the input buffer will use excessive power.
  - Digital input buffers can be disabled by writing to the digital input disable registers (**DIDR1** and **DIDR0**).

0.9 RESETTING AVR

All I/O registers are set to their intial values and program starts execution form reset vector.

`resetLogic.png`

0.9.1 Reset Sources

(I) Power-on Reset - MCU resets when supply voltage is below the power-on reset threshold(V_{POT}).

(II) External Reset - MCU resets when low level is present on $\overline{RESET}$ is helow for minimum pulse length.

(III) Watchdog System reset - MCU resets when watchdog timer period expires and watchdog system reset mode is enabled.

(IV) Brown-out reset - MCU resets when supply voltage V_{CC} is below brown-out threshold (V_{BOT}) and brown-out detected is enabled.

Power-on Reset

POR1.png

MCU Start-up, $\overline{RESET}$ Tied to V_{CC}

POR2.png

MCU Start-up, $\overline{RESET}$ Extended Externally

- A power-on reset (POR) pulse is generated by an on-chip detection circuit.
- The POR is activated whenever V_{CC} is below the detection level.
- The POR circuit can be used to trigger the start-up reset, as well as to detect a failure in supply voltage.
- Reaching the power-on reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise.

External Reset

externalReset.png

- An external reset is generated by a low level on the $\overline{RESET}$ pin.
- Shorter pulses are not guaranteed to generate a reset.

- When the applied signal reaches the reset threshold voltage – V_{RST} – on its positive edge, the delay counter starts the MCU after the time-out period – t_{OUT} – has expired.
- The external reset can be disabled by the ***RSTDISBL*** fuse.

Brown-out Detection

brownOutReset.png

- On-chip brown-out detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level.
- The trigger level for the BOD can be selected by the ***BODLEVEL*** fuses.

Watchdog System Reset

watchDogReset.png

- When the watchdog times out, it will generate a short reset pulse of one CK cycle duration.
- On the falling edge of this pulse, the delay timer starts counting the time-out period t_{OUT}.