EE6304 - EMBEDDED SYSTEMS DESIGN

ASSIGNMENT 01

NAME : SAIAMIRTHAN A.T.

REG. NO. : EG/2020/4173

SEMESTER : 06

DATE : 20/10/2024

1 BOOTLOADER

1.1 BOOTLOADER IN EMBEDDED SYSTEMS

Once we power up an embedded system device the first piece of code that is powered up is known as the bootloader. The bootloader first loads up the application into memory and initiates it, the size of it is about 1KB of memory stored in flash memory for the device to function immediately.

This piece of code consists of a main function, interrupts and is in the executable. hex format. This is what allows us to program our ATMega328p over a USB-cable. This bridges the gap between the hardware and software. The bootloader depends on several factors as well such as the computer architecture and the configuration of the boards as well.

A device with no bootloader cannot be used to fix bugs, improve performances etc. Making the device not changeable. But a device can function without a bootloader as well. The bootloader can be used to perform software updates and this is important to fix bugs, improvements security and performance.

Bootloader is also very important when updating the device firmware by verifying the integrity of the new firmware. This will also hold a previous working version as a backup in case of errors.

The bootloader in embedded systems is essential for initializing hardware before the system fully starts. When power is applied, the microprocessor runs bootloader that prepares the system by initializing the peripherals and making them ready for operation. Just like in a PC, the bootloader introduces the microprocessor to its connected devices, ensuring everything is set up correctly for further execution.

1.2 BOOTLOADER FUNCTION IN THE ATMEGA328P

The bootloader located in the flash memory, the size of the bootloader depends on the BOOTZ fuse settings and it can range on between 256kB to 2KB. The bootloader is executed first and it initializes the peripherals and checks for system errors. Once a new firmware is detected by the use of serial communication protocols it transfers to the main flash. Once the new firmware is updated the application code stored at the 0x0000 takes control and the main program runs

The bootloader also facilitates us to update the firmware while the device is in a circuit without the need for it to be removed. (i.e. known as In-System Programming)

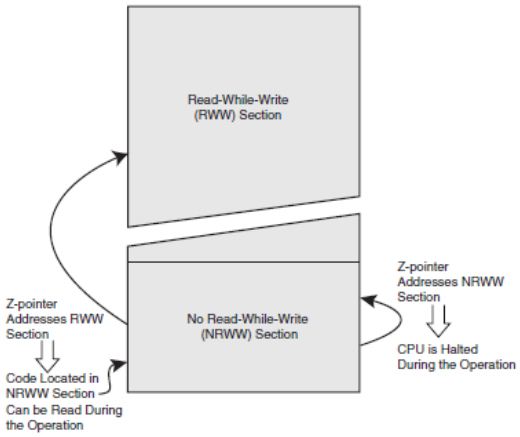

Figure 1: Read while Write vs No read while Write

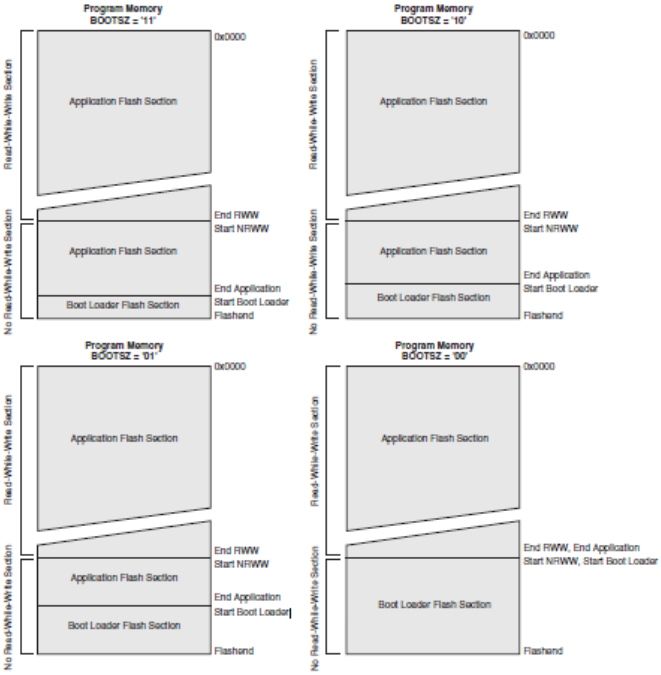

Figure 2: Memory Section

1.3 STEPS INVOLVED IN FLASHING A PROGRAM USING A BOOTLOADER

First the device has to be powered on, depending on the fuse settings it goes to the bootloadder section, then a suitable serial communication protocol is established and the program is recieved through a .hex file from us , then this is written into the flash memory by the bootloader and verified when no errors are detected the system is reset and the application code is executed.

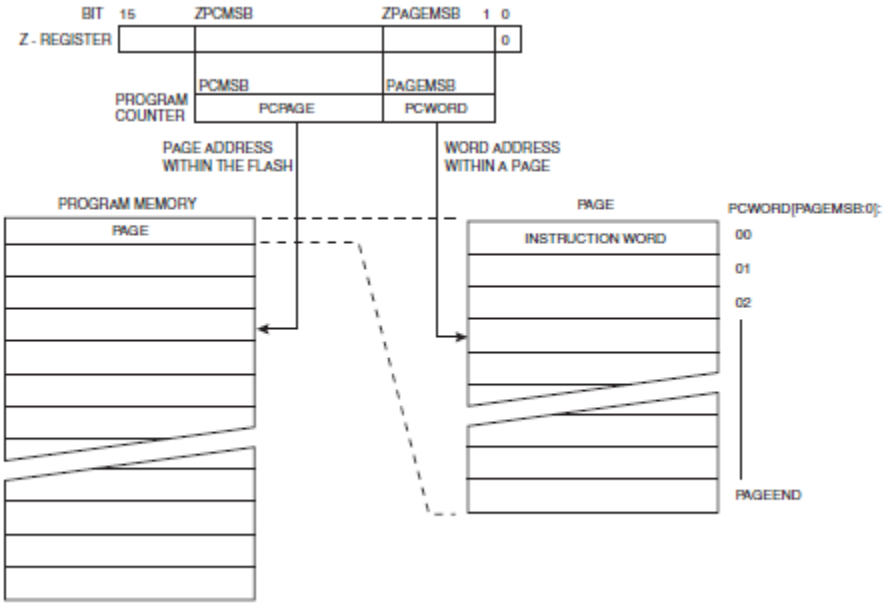

Figure 3: Addressing the Flash during SPM

2 WATCHDOG TIMER

2.1 PURPOSE OF THE WATCHDOG TIMER IN EMBEDDED SYSTEMS

As the name implies this is implemented to overlook the system to ensure the stability and reliability. In embedded systems devices various interferences and failures can occur due to external environmental factors or internal factors so what a watch dog timer basically does is detect any anomalies in the normal mode of operation and take steps to restore it.

This is basically a timer which resets after the system boots and times in accordance to a time out period. The system has to feed the dog within this timeout else it generates a response such as system reset of shutdown, as the system is deemed to be faulty.

2.2 HOW IT'S IMPLEMENTED IN ATMEGA 328P

The watchdog timer basically works by resetting the timer periodically indicating no errors in the system and it does so in AtMega328P by the following steps.

The WDTCSR(WatchDog Timer Control Register) register controls the configurations,timeout,reset modes and interrupts.

The timeout periods vary from 8ms to 16 ms and we adjust the relevant pre-scalar accordingly.

WDE(WatchDog System Reset Enable) bit is set to enable the Watchdog timer reset, in the event that the WDE is set it'll automatically system reset after a timeout. The WDP (WatchDog Pre-scaler) bits are set for the timeout.

WDP 0000 indicates that 16ms is the reset time whereas WDP 1111 indicates the timeout is 8ms.

By configuring the WDT register an interrupt can be made when a timeout happens. The WDTCSR register also made that no easy modification is done.

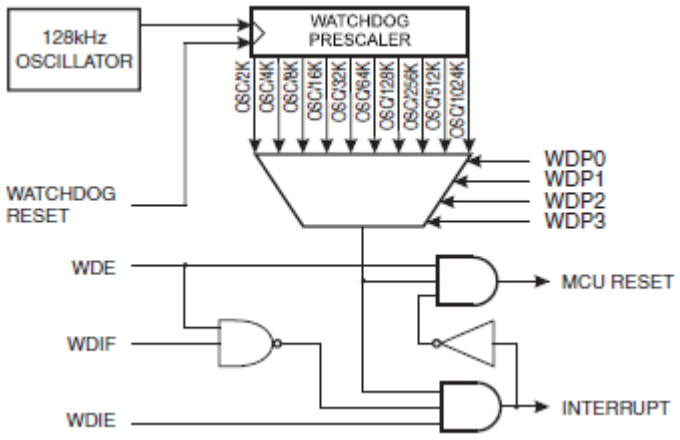

Figure 4: Watchdog Timer

2.3 DISCUSS HOW WATCHDOG TIMER IN ATMEGA 328P HELPS IMPROVE SYSTEM RELIABILITY AND WHAT HAPPENS WHEN IT RESETS THE SYSTEM.

The watchdog timer is critical in different aspects in a ATmega 328P application .It helps detection of system failures by resetting this watchdog timer. Failing to do so will result in infinite loop, exception handlers etc.

For example, if we consider we use this microcontroller to measure the real-time heartbeat of a person and in the event, this gets stuck in an infinite loop, the system will not respond hence the WDT will help the system to not get frozen.

In such an event as above where the device remains unresponsive the WDT automatically triggers a system reset after the timeout allowing the device to restart its operations. This in turn minimizes the system downtime as well ensuring the time the system is unresponsive is a minimum.

The MCU is reset when the timer expires and the watchdog system reset mode is enabled. Then the device acts like it's been turned on. Then the bootloader begins executing by re-initializing the peripherals, timer states etc. Once the system reset occurs the Watchdog Reset Flag in the MCUSR register is set.

3 CODE IMPLEMENTATION

3.1 Example Code

```
WDT_on:
    ; Turn off global interrupts
    cli
    ; Reset Watchdog Timer
    wdr
    ; Write logical one to WDCE and WDE
    ; This allows changes to the WDTCSR register
    lds r16, WDTCSR
```

```
ori r16, (1<<WDCE) | (1<<WDE)
sts WDTCSR, r16
; Set the desired pre-scaler for the timeout period
;we set WDP to give a 16ms timeout
ldi r16, (1<<WDE) | (1<<WDP0)
sts WDTCSR, r16
; Turn on global interrupts
sei
ret
```

3.2 How the Bootloader Works

Once the watchdog timer performs a reset, the MCU proceeds to the bootloader memory, then the bootloader waits for a serial communication from and external device, this is established to receive firmware updates. The bootloader erases the previous code verifies it and then runs to the application code.

3.3 Importance for Firmware Updates

Every time a new firmware update is given the bootloader checks for validity and verifies to prevent faulty code it also stores a previous version of the working code. The bootloader also allows updating firmware without the need for specialized devices and allows for feature additions and bug fixes without the need for reprogramming or disassembling the device.

4 Conclusion

The bootloader and the watchdog timer are important devices in controlling the stability and functionality of the system and device. These are critical functions in real world applications where reliability and real time updates are essential.

Let s consider an airbag system, the watchdog timer plays a crucial role in ensuring reliability and responsiveness. The system being unresponsive during a car accident may compromise the

safety of the induvial. The timer detects any anomalies and resets the device to ensure normal operation and prevent fatal errors.

In a medical device such as a life support system the monitoring and continuous operation is critical. In the event that these devices need bug fixes, security fixes or even feature updates the bootloader will help receive firmware updates without disconnecting the device. This ensures that the device is up to date and is operating continuously. This is very crucial for the patient's wellbeing

5 REFFERENCE

[1] Microchip Technology, “ATmega328P: 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash,” Microchip Technology Inc., 2020. [Online]. Available: https://www.microchip.com. [Accessed: Oct. 20, 2024].

[2] Electronics Basics, "What is a Watchdog Timer?," *YouTube*, Oct. 4, 2020. [Online]. Available: https://www.youtube.com/watch?v=r6u9KTgkrzs&t=137s. [Accessed: Oct. 20, 2024].