ECE 375 Prelab 4

Writing Assembly Code - Prelab

Lab Time: Thursday 1000-1200

Eric Prather

PRELAB QUESTIONS

1. The stack pointer is a register that stores a memory address pointing to the location on a first-in first-out (FIFO) region of memory called the “stack”. It can be used to keep track of a call stack or for data-although in this lab, its main purpose should be as an augment to the program counter (PC). It is initialized in the SPI_INIT: section of our AVR code to the manual-prescribed memory address (see manual page 13).

   Pseudocode:

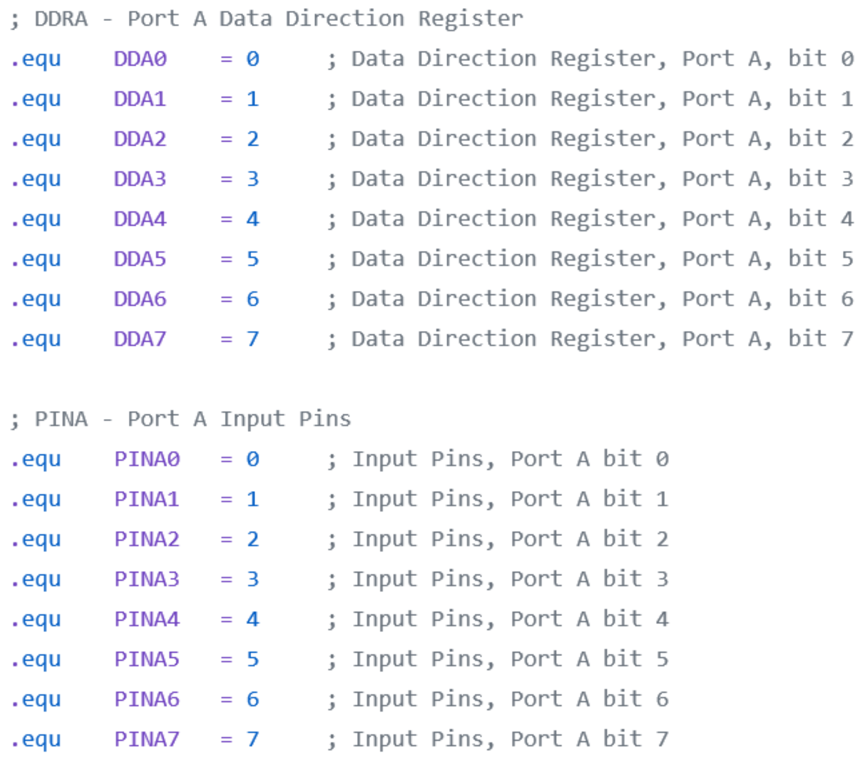
   ```
   Initialization Region:
           R16 = High byte of RAMEND // const
           Stack Pointer 1 = R16
           R16 = low byte of RAMEND  // const
           Stack Pointer high = r16
           //Cannot combine instructions
   ```

2. The AVR instruction LPM is short for “Load Program Memory”. It loads the value pointed to by Z to the specified register. It is different from the regular load command, which loads from data memory, not program memory. Program memory is organized in 2B words. Here is some pseudocode to demonstrate its operation:

   Pseudocode:

   r17:16 = loadProgramMemory(Z) // r17:16 = programMemory[Z]

3. This assembly definition file provides a series of useful macros, similar to the atmel c headers (Discussed in Prather_Eric_Lab2_Report.pdf). For example, the I/O pins are all given names and values here via .equ commands. Some other important features, such as the special registers and special data pointers, are also defined here. This question requires examples of definitions to be provided, so here is a screenshot:



```
; DDRA - Port A Data Direction Register
.equ    DDA0    = 0     ; Data Direction Register, Port A, bit 0
.equ    DDA1    = 1     ; Data Direction Register, Port A, bit 1
.equ    DDA2    = 2     ; Data Direction Register, Port A, bit 2
.equ    DDA3    = 3     ; Data Direction Register, Port A, bit 3
.equ    DDA4    = 4     ; Data Direction Register, Port A, bit 4
.equ    DDA5    = 5     ; Data Direction Register, Port A, bit 5
.equ    DDA6    = 6     ; Data Direction Register, Port A, bit 6
.equ    DDA7    = 7     ; Data Direction Register, Port A, bit 7

; PINA - Port A Input Pins
.equ    PINA0   = 0     ; Input Pins, Port A bit 0
.equ    PINA1   = 1     ; Input Pins, Port A bit 1
.equ    PINA2   = 2     ; Input Pins, Port A bit 2
.equ    PINA3   = 3     ; Input Pins, Port A bit 3
.equ    PINA4   = 4     ; Input Pins, Port A bit 4
.equ    PINA5   = 5     ; Input Pins, Port A bit 5
.equ    PINA6   = 6     ; Input Pins, Port A bit 6
.equ    PINA7   = 7     ; Input Pins, Port A bit 7
```