Embedded Systems
Professor Thomas Bräunl

Tutorial 4 – From Assembly to C + Gates SOLUTIONS

1. (a) From the following fragment of assembly code, complete the table:

```
1.  LDS  R16, $0050
2.  LDI  R17, $51
3.  STS  $004A, R16
4.  STS  $004B, R17
```

Registers and Mem.	Initial Values	After 1.	After 2.	After 3.	After 4.
(PC)	$00	$02	$03	$05	$07
(R16)	$00	$42	$42	$42	$42
(R17)	$FF	$FF	$51	$51	$51
($004A)	$3C	$3C	$3C	$42	$42
($004B)	$1D	$1D	$1D	$1D	$51
($0050)	$42	$42	$42	$42	$42
($0051)	$B9	$B9	$B9	$B9	$B9

(b) What is the value in the following registers and/or memory locations after executing the following instructions?

1. LDS R16, $0400

Before	*After*
R16 =$76	R16 = *$89*
[$0400] =$89	[$0400] = *$89*

2. LDI R16, $04

Before	*After*
R16 =$76	R16 = *$04*
[$0400] =$89	[$0400] = *$89*

3. CPI R16, $76

Before	*After*
R16 =$76	R16 = *$76*
[$0400] =$89	[$0400] = *$89*

-- Flags: Overflow, Negative, Zero, Carry = _ _ _ _--
-- 0010
Note: The CPI instruction performs subtraction but does not store the result anywhere. Instead it sets the flags in the Condition Code Register.

4. LDS R16, $0400
STS $0401, R16
Before *After*
R16 =$76 R16 = *$89*
[$0400] =$89 [$0400] = *$89*
[$0401] =$00 [$0401] = *$89*

5. ADD R16, R17
Before *After*
R16 =$76 R16 = $88
R17 =$12 R17 = $12

6. AND R16, R17
Before *After*
R16 =$76 R16 = 0111 0110 AND 0001 0010 = $12
R17 =$12 R17 = $12

7. OR R16, R17
Before *After*
R16 =$76 R16 = 0111 0110 OR 0001 0010 = $76
R17 =$12 R17 = $12

8. INC R30
Before *After*
R30 =$79 R30 = $7A

9. DEC R30
Before *After*
R30 =$00 R30 = $FF

10. CLR R30
Before *After*
R30 =$FF R30 = $00

2. Translate the following Assembly program into C

```
; Project: Moving LEDs
.include "m169def.inc"

main: LDI R16, 0xFF   ; D is output
      OUT DDRD, R16
      LDI R16, 1      ; init count

loop: OUT PORTD, R16  ; display LED
      CALL wait
      LSL R16         ; shift left
      BRNE loop
      LDI R16,1           ; if 0 -> 1
      JMP loop

wait:      LDI  R31, 255 ; init cnt
waitloop:  DEC  R31
           BRNE waitloop
           RET
```

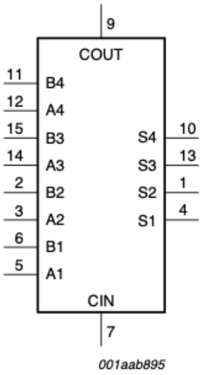
```
int count;

void setup()
{ pinMode(1, OUTPUT);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  count = 0b001;
}

void wait()
{ int i;
  for (i=255; i>0; i--) ; /* wait */
}

void loop()
{ digitalWrite(1, count & 0b100); // highest bit
  digitalWrite(2, count & 0b010);
  digitalWrite(3, count & 0b001); // lowest bit
  count = count << 1;  // shift left
  if (count > 0b100) count = 0b001;
  wait();
}
```

3. Review the Gates required for buildign a hardware Adder

(a) Half-Adder requires 1 x AND and 1 x XOR.
How can you build this circuit using only NAND and NOR gates?

(b) How do you connect an LED to a chip output?
Do you need a resistor and if so, which size?

(c) Review the 74HC283 Adder chip. Which additional pats do you need to build a 2 x 4-bit adder?



No solution given. Ou just need to build this in hardware in the lab.