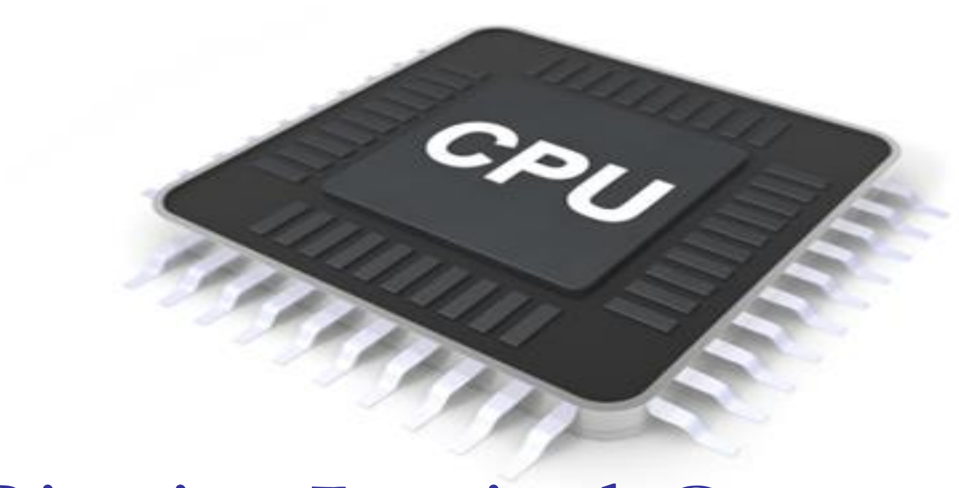


Bitwise Logical Operations

ENEE 3582

Microp

Bitwise AND

- Bitwise AND:
  - $0 \cdot 0 = 0$
  - $0 \cdot 1 = 0$
  - $1 \cdot 0 = 0$
  - $1 \cdot 1 = 1$
- AND
  - Syntax: `AND Rd,Rs    ;Rd = Rd · Rs`
  - `Rd,Rs` can be `R0…R31`
- ANDI
  - Syntax: `ANDI Rd,K    ;Rd = Rd · K`
  - `Rd,Rs` can be `R16…R31`
  - `K` = 0…255

Bitwise OR

- Bitwise OR:
  - 0 v 0 = 0
  - 0 v 1 = 1
  - 1 v 0 = 1
  - 1 v 1 = 1
- OR
  - Syntax: `OR Rd,Rs    ;Rd = Rd v Rs`
  - `Rd,Rs` can be `R0…R31`
- ORI
  - Syntax: `ORI Rd,K    ;Rd = Rd v K`
  - `Rd,Rs` can be `R16…R31`
  - `K` = 0…255

Logical Bitwise Ex-OR

- Bitwise Ex-OR:
  - $0 \oplus 0 = 0$
  - $0 \oplus 1 = 1$
  - $1 \oplus 0 = 1$
  - $1 \oplus 1 = 0$

- Syntax: `EOR Rd,Rs` `;Rd = Rd ⊕ Rs`
- `Rd,Rs` can be `R0…R31`

Clear

- CLR
  - Clear a register
  - Syntax: `CLR Rd ;Rd = 0`
  - Rd can be R0…R31
- CBR
  - Clear a bit(s) in register
  - Syntax: `CBR Rd,K ;clear bits in Rd that are 1 in K`
  - Rd can be R16…R31
  - K = 0…255
  - Example:

```
LDI R16, 0xff           ;R16=0b11111111
CBR R16, 0b11110000     ;R16=0b00001111
```

Set

- SER
  - Set a register
  - Syntax: `SER Rd          ;Rd = 0b11111111 = 0xFF`
  - Rd can be R…R31
- SBR
  - Set a bit in a register
  - Syntax: `SBR Rd,K        ;Rd = Rd v K`
  - Rd can be R16…R31
  - K = 0…255
  - Example:

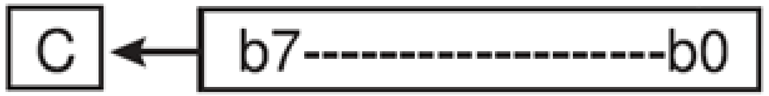
```
LDI R16, 0                 ;R16=0b00000000
SBR R16, 0b11110000        ;R16=0b11110000
```

Complement and Negate

- COM
  - Complement a reg (aka 1's complement)
  - Syntax: `COM Rd      ;Rd = not(Rd)`
- NEG
  - Negate a register (aka 2's complement)
  - Syntax: `NEG Rd      ;Rd = -Rd`

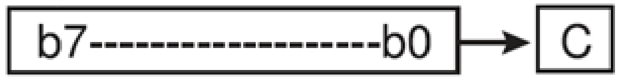
Logical Shift Left

- Logical Shift Left Once
- CF = msb
- Syntax: `LSL   Rd          ;Rd = Rd << 1`
- Rd can be `R0…R31`
- Similar to multiplying by 2

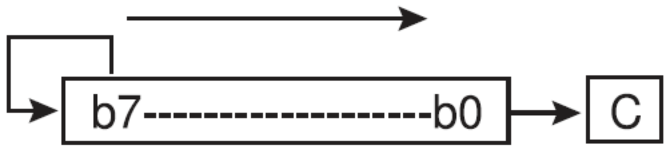


Shift Right

- Similar to dividing by 2
- LSR
  - Logical Shift Right Once
  - CF = lsb
  - Syntax: `LSR Rd ;Rd = Rd >> 1`
  - Rd can be `R0…R31`

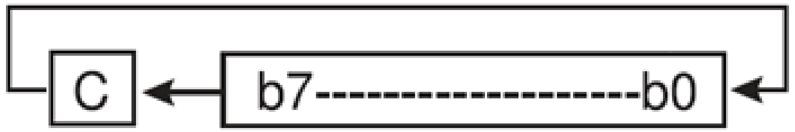


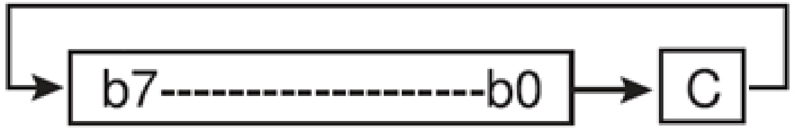
- ASR
  - Arithmetic Shift Right Once
  - CF = lsb
  - Syntax: `ASR Rd ;Rd = Rd >> 1`
  - Rd can be `R0…R31`
  - Replicates the most significant bit



Rotate

- ROL
  - Rotate Left Once through Carry
  - Syntax: `ROL Rd`
  - Rd can be `R0…R31`
- ROR
  - Rotate Right Once through Carry
  - Syntax: `ROL Rd`
  - Rd can be `R0…R31`





Examples of Shift

```
LDI R16, 0b00110011  ;                          0b00110011= 51
LSL R16              ;R16=01100110, CF=0        0B01100110=102
LSL R16              ;R16=11001100, CF=0        0B11001100=204
LSL R16              ;R16=10011000, CF=1        0B10011000=152 OR -104
ASR R16              ;R16=11001100, CF=0        0B11001100=-52
LSR R16              ;R16=01100110, CF=0
ASR R16              ;R16=00110011, CF=0
ROR R16              ;R16=00011001, CF=1
ROL R16              ;R16=00110011, CF=0
```