EE5314 Class Notes

33FJ128MC802 Instruction Set

Spring 2011

Dr. Jason Losh

Instruction Set Topics

- Opcode Symbols
- Instruction Set Groups

Resources

- Resources on the Class Server
  - Microchip 33FJ128MC802 Datasheet
  - Microchip 33F Family Reference Manual
  - Microchip 16-bit MCU and DSC Programmer’s Reference Manual

Miscellaneous Symbols

- #literal
- [ ] the contents of
- { } optional
- <x:y> bit in a number
- Expr label or absolute address)
- .b byte modifier
- .w word modifier (default)
- .d double word modifier
- .s shadow register modifier

Modifier Symbols

- Multiplier
  - .ss signed, signed for multiply
  - .su signed, unsigned for multiply
  - .us unsigned, signed for multiply
  - .uu unsigned, unsigned for multiply
- Divider
  - .sw signed, 16 numerator/16-bit denominator
  - .uw unsigned, 16 numerator/16-bit denominator
  - .sd signed, 32 numerator/16-bit denominator
  - .ud unsigned, 32 numerator/16-bit denominator

Literal, Bitfield, and Reg Symbols

- Bit4 4-bit bit selection field
- C, N, OV, Z (flags)
- PC (program counter)
- Expr (label or absolute address)
- f (file register address (0-0x1FFF)
  - Lower 8192 bytes of data memory
- lit*x* (x-bit constant)
- slit*x* (x-bit signed constant)

Working Regs in Instructions

- WREG name for W0 in file reg instructs
- Wb base register
- Wm, Wn dividend, divisor reg pair
- Wn register
- Wnd destination register
- Wns source register

Working Regs in Instructions

- Wd destination which can be Wnd or [Wnd] (w/ optional post/pre inc/dec mods)
- Ws source which can be Wns or [Wns] (w/ optional post/pre inc/dec mods)
- Wdo destination which can be Wnd, [Wnd] (w/ optional post/pre inc/dec mods), or [Wnd + Wb]
- Wso source which can be Wns, [Wns] (w/ optional post/pre inc/dec mods), or [Wns + Wb]

Byte Width Usage

- In general, most commands work with the .B modifier
- Notable exceptions:
  - MOV f, Wnd and MOV Wns, f
    (use MOV.B f, WREG and MOV.B WREG, f instead)
  - EXCH Wns, Wnd (broken in silicon, do not use)
  - MUL.SS, MUL.US, MUL.SU, MUL.UU
    (use MUL.B f instead)
  - ASR/LSR/SL Wb, #lit4 and Wns, Wnd
  - FF1L, FF1R
  - PUSH and POP

Double-word Width Usage

- Very few commands support
- These commands do:
  - MOV.D
  - MUL.__ stores 16bx16b result in 32b
  - DIV.SD and DIV.UD use 32b numerator

File Register Instructions

- Used for 18F backwards compatibility
- File register (f) is a 13-bit field, so accesses near memory (first 8192 bytes)
- Instructions of the form
  - OP f, WREG
    WREG = f OP WREG
  - OP f or OP f, f
    f = f OP WREG

Move Operations

- Literals
  - MOV #lit16, Wn
  - MOV.B #lit8, Wn
- Direct (Near Memory)
  - MOV f, Wn (no byte ops)
  - MOV Wn, f (no byte ops)
  - MOV f {, WREG}
  - MOV WREG, f

Move Operations

- Direct, Indirect, and Indexed Offset
  - MOV [Wns + slit10], Wnd
  - MOV Wns, [Wnd + slit10]
  - MOV Wso, Wdo
- Double-word reg-to-reg and indirection
  - MOV.D Wns, Wd
  - MOV.D Ws, Wnd

Move Operations

- Exchange
  - EXCH Wns, Wnd
    (Note: No indirection, no byte operations)
- Swap
  - SWAP{.B} Wn; swap LSN and MSN
  - SWAP Wn; swaps LSB and MSB

Arithmetic Operations

- ADD, ADDC (with carry), SUB, SUBB (with borrow), SUBR (reverse), SUBBR (reverse with borrow), AND, IOR, XOR
  - OP f, {WREG}
  - OP #lit10, Wn (not for SUBR, SUBBR)
  - OP Wb, Ws, Wd
  - OP Wb, #lit5, Wd
- DEC, DEC2, INC, INC2, NEG (2’s compl)
  - OP f, {WREG}
  - OP Ws, Wd

Arithmetic Operations

- MUL.SS, MUL.SU, MUL.US, MUL.UU (signed, unsigned permutations)
  - MUL.__ Wb, Wd, Wnd
- MUL.SU, MUL.UU
  - MUL.__ Wb, #lit5, Wnd
- MUL f (W3:W2 = f * WREG)
- DIV.SW, DIV.UW, DIV.SD, DIV.UD (signed/unsigned, word double word permutes)
  - DIV.__ Wm, Wn
    (Note: Stores int result and remainder)

Conversion and Fill Operations

- SE (sign extend byte into word), ZE (zero extend byte into word)
  - OP Ws, Wnd
    (Note: No indirection in destination)
- CLR, SETM (clear or set all bits)
  - OP f
  - OP WREG
  - OP Ws
- DAW.B (binary to BCD)
  - OP Wn

Logical Operations

- AND, IOR, XOR
  - OP f, {WREG}
  - OP #lit10, Wn
  - OP Wb, Ws, Wd
  - OP Wb, #lit5, Wd
- COM (1’s compliment)
  - OP f, {WREG}
  - OP Ws, Wd

Shift and Rotate Operations

- ASR (signed), LSR (unsigned), SL
  - – OP f, {WREG}
  - – OP #lit10, Wn
  - – OP Wb, Wns, Wnd (no byte operations)
  - – OP Wb, #lit5, Wnd (no byte operations)
- RLC, RLNC, RRC, RRNC
  - – OP f, {WREG}
  - – OP Ws, Wd

Bit Operations

- BCLR (clear), BSET (set), BTG (toggle)
  - OP f, #bit4
  - OP Ws, #bit4
- BSW.C, BSW.Z (set bit Wb if flag set)
  - BSW._ Ws, Wb
- FF1L, FF1R, FBCL (find first one bit or bit change from left or right, store in Wnd, no byte operations)
  - OP Ws, Wnd

Bit Operations

- BTST (test bit, store in Z)
  - OP f, #bit4 (BTST)
- BTST.C, BTST.Z (test bit, store in C or Z), BTSTS.C, BTSTS.Z (test bit, store in C or Z, then set bit)
  - OP Ws, #bit4
  - OP Ws, Wb

    (Note: BTSTS very useful for coding semaphores and mutexs)

DSP Operations

- LAC (load accumulator)
  - LAC Ws, {#Slit4,}, Acc
    (ACCx = Extend(Ws) >> Slit4)
- SAC[.R] (store accumulator with rnd opt)
  - SAC Acc, {#Slit4,}, Wd
    (Wd = Acc [31..16] >> Slit4
- NEG (negate accumulator)
  - NEG Acc

DSP Operations

- ADD/SUB Acc (add/sub accumulator)
  - ADD/SUB Acc (ACCx <-ACCA + ACCB)
  - ADD/SUB Ws, {#Slit4,} Acc
    (ACCx = ACCx +/- Extend(Ws) >> Slit4)
- SFTAC (shift accumulator by literal or (Wb))
  - SFTAC Acc,#Slit6
    Acc = Acc >> #Slit6
  - SFTAC Acc, Wb
    Acc = Acc >> Wb

DSP MAC Operations

- CLR (clear accumulator)
- ED and EDAC (Euclidean distances)
- MAC (multiply and add to accumulator)
- MSC (multiply and add to accumulator)
- MPY[.N] (multiply to accumulator
- MOVSAC (move from X and Y bus)

- Details in later examples

Table Operations

- TBLRDH (read LSW), TBLRDL (read MSW), TBLWTL (write LSW), TBLWTH (write MSW)
  - OP Ws, Wd
    (Note: .B modifier reads/writes MSB or LSB or selected word)

Stack Operations

- LNK #lit14 (push W14, stores W15 in W14, reserves #lit14 bytes for local variables, adjusts W15)
- ULNK (undoes the LNK instruction)
- PUSH (no byte operations)
  - OP f
  - OP Wso (push), OP Wdo (pop)
  - OP Wns (push, OP Wso (pop)
  - OP.S

Test Operations

- CP (subtract without destination write), CPB (subtract with borrow without destination write)
  - CP f
  - CP Wb, #lit5
  - CP Wb, Ws
- CP0 (compare with zero)
  - CP0 f
  - CP0 Ws

Conditional Flow Control Ops

- BTSC, BTSS
  (skip next instruction if bit clear or set)
  - OP f, #bit4
  - OP Ws, #bit4
- CPSEQ. CPSGT, CPSLT, CPSNE
  (skip next instruction if cond true)
  - OP Wb, Wn

Conditional Flow Control Ops

- REPEAT (repeat next instruction N times)
  - REPEAT #lit14
  - REPEAT Wn
- BRA cond ((jump -32768 to +32767 program words relative to next instruction address)
  cond = C (carry), NC, Z (zero), NZ, OV (overflow), NOV, N (negative), NN;
  GE, GT, LT, LE (signed);
  GEU, GTU, LTU, LEU (unsigned)
  - BRA cond, Expr
    (Note: To use DC, use bit test of SR<8>)

Unconditional Flow Control Ops

- GOTO, CALL (anywhere in program memory)
  - GOTO Expr
  - GOTO Wn
- BRA, RCALL (-32768 to +32767 program words relative to next instruction address)
  - BRA Expr
  - BRA Wn

Unconditional Flow Control Ops

- RETURN (return)
- RETLW #lit10, Wn (return with value)

Interrupt and Control Operations

- RETFIE (returns from interrupt)
- DISI #lit14 (disable level 0-6 interrupts for #lit14 cycles if DISI flag (INTCON<14>) is set)
- NOP (inserts a no operation cycle)
- CLRWDT (clears watchdog timer)
- PWRSAV (puts processor in power saving state)
- RESET (resets processor)