ECE 471 – Embedded Systems
Lecture 10

Vince Weaver

http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

23 September 2020

Announcements

- HW#3 due Friday
- Don't forget to pick up parts (outside Barrows 203 M/W/F)
- If you don't have a breadboard let me know

HW2 Review

- Most people seem to be accessing the Pi OK
- Be sure to follow directions!
- Put your name in the README
- Testing. How can you test? `wc -l`
- Watch for off-by-one errors
- Comment your code!
- Also watch out for compiler warnings! (Though each compiler version might have different warnings)
- Error handling! especially for command line parsing

- Most C code OK.
  Be sure if it says print 12 lines that you do, not 13.
  Colors seem not to be a problem.

- more info on ls. Looking for man. “info” or `ls --help`
- ls -a shows hidden files. Hidden files on UNIX
- Why use C? close to hardware, easier than assembly, etc.

ARM Instruction Set Encodings

- ARM – 32 bit encoding
- THUMB – 16 bit encoding
- THUMB-2 – THUMB extended with 32-bit instructions
  - STM32L *only* has THUMB2
  - Original Raspberry Pis *do not* have THUMB2
  - Raspberry Pi 2/3 *does* have THUMB2
- THUMB-EE – extensions for running in JIT runtime
- AARCH64 – 64 bit. Relatively new. Completely different from ARM32

THUMB-2

- Extension of THUMB to have both 16-bit and 32-bit instructions
- The 32-bit instructions are *not* the standard 32-bit ARM instructions.
- Most 32-bit ARM instructions have 32-bit THUMB-2 equivalents *except* ones that use conditional execution. The `it` instruction was added to handle this.
- `rsc` (reverse subtract with carry) removed
- Most cannot have PC as src/dest

- Shifts in ALU instructions are by constant, cannot shift by register like in arm32
- THUMB-2 code can assemble to either ARM-32 or THUMB2

  The assembly language is compatible.

  Common code can be written and output changed at time of assembly.
- Instructions have "wide" and "narrow" encoding.

  Can force this (`add.w` vs `add.n`).
- Need to properly indicate "s" (set flags).

  On regular THUMB this is assumed.

New THUMB-2 Instructions

- BFI – bit field insert
- RBIT – reverse bits
- movw/movt – 16 bit immediate loads
- TB – table branch
- IT (if/then)
- cbz – compare and branch if zero; only jumps forward

Thumb-2 12-bit immediates

```
top 4 bits 0000 -- 00000000 00000000 00000000 abcdefgh
           0001 -- 00000000 abcdefgh 00000000 abcdefgh
           0010 -- abcdefgh 00000000 abcdefgh 00000000
           0011 -- abcdefgh abcdefgh abcdefgh abcdefgh
rotate bottom 7 bits|0x80 right by top 5 bits
           01000 -- 1bcdefgh 00000000 00000000 00000000
            ...
           11111 -- 00000000 00000000 00000001 bcdefgh0
```

IT (If/Then) Instruction

- Allows limited conditional execution in THUMB-2 mode.
- The directive is optional (and ignored in ARM32) the assembler can (in-theory) auto-generate the IT instruction
- Limit of 4 instructions

Example Code

```
it cc
addcc r1,r2
```

```
itete cc
addcc r1,r2
addcs r1,r2
addcc r1,r2
addcs r1,r2
```

AARCH64

- 32-bit fixed instruction encoding
- 32 64-bit GP registers
  - x0 - x7 = args
  - x8 - x18 = temp (x8=syscall num during syscall)
  - x19-x28 = callee saved
  - x29 = frame pointer
  - x30 = link register
  - x31 = zero register or stack pointer
- PC is not a GP register

- only branches conditional
- no load/store multiple
- No thumb

Final Code Density

mention ll project