ECE 471 – Embedded Systems
Lecture 25

Vince Weaver

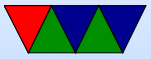
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

3 November 2021

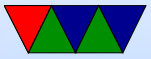
Announcements

- Don't forget HW#8
- Don't forget Project topics
  will respond to them via e-mail



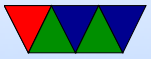
HW#7 Review – Code

- People managed merging 2 bytes into one OK
- A lot of trouble converting binary to hexadecimal.
  Trouble with embedded systems is off-by one values like this can be really hard to debug
  Note on gcc at least you can enter binary constants like `0b10100101`
- Divide by 1023 vs 1024
- What is the max frequency? Last year someone setting to 500kHz by accident, a few degrees different. Data



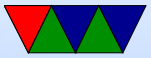
sheet unclear

-
- Errors: exiting. Not print plausibly real invalid values. In our case, printing 0V when actually 3.3V not an issue, but imagine if it were 10,000V and you print 0V



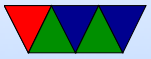
HW#7 Review – Questions

- Anti-lock brakes hard/soft/firm realtime?
  Hard. If things go wrong would be disaster
- Stereo change channel hard/soft/firm realtime?
  Soft. Prefer it not to be late, but still want to happen
- Video coming in at 60fps decoding?
  Firm, if frame decoded late it is useless
- Disadvantage of SPI?
  More wires, no standard, no errors
- Advantage of SPI?



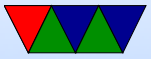
Lower Power, Full Duplex, No max speed

- TMP36 on end of cable.
  Voltage Drop, Noise?
  Datasheet has two options, convert to current, or an extra resistor.
- Minimum frequency of 10kHz or results invalid. Maybe cannot go this fast if bitbanging via GPIO. Also context switch in middle, Linux not realtime?



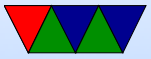
HW#7 Review – Linux "fun"

- /dev/null
- /dev/full
- /dev/zero, holes in files
- /dev/random – give explanation on sources of randomness (entropy), pseudo-randomness, etc.
- Mention related DOS/Windows compatibility issue with device filenames



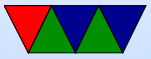
Types of Security Compromise

- Crash
  "ping of death"
- DoS (Denial of Service)
- User account compromise
- Root account compromise
- Privilege Escalation
- Rootkit
- Re-write firmware? VM? Above OS?



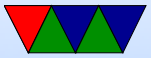
Unsanitized Inputs

- Using values from users directly can be a problem if passed directly to another process
- If data (say from a web-form) directly passed to a UNIX shell script, then by including characters like ; can issue arbitrary commands: `system("rm %s\n",userdata);`
- SQL injection attacks; escape characters can turn a command into two, letting user execute arbitrary SQL commands; xkcd `Robert '); DROP TABLE Students;--`



Buffer Overflows

- User (accidentally or on purpose) copies too much data into a fixed sized buffer.
- Data outside expected area gets over-written. This can cause a crash (best case) or if user carefully constructs code, can lead to user taking over program.

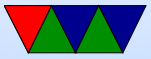


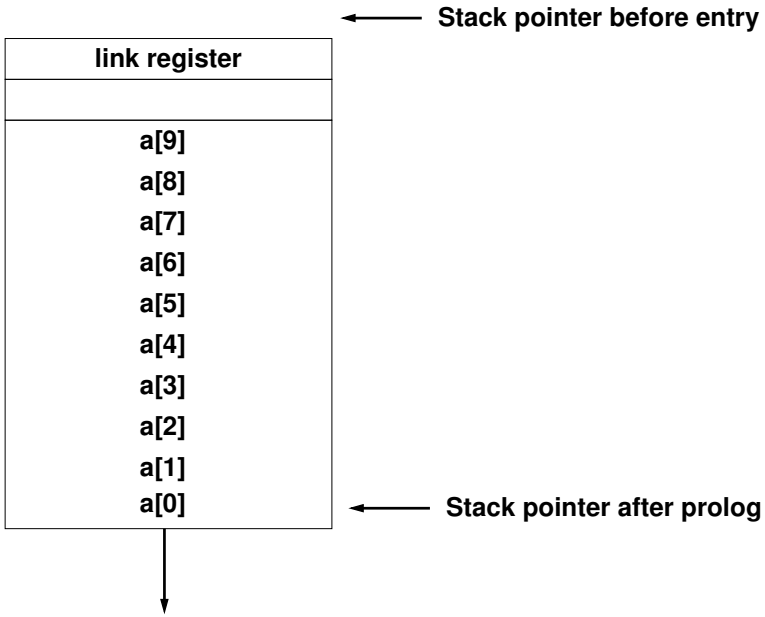
Buffer Overflow Example

```
void function(int *values, int size) {
    int a[10];

    memcpy(a,values,size);

    return;
}
```

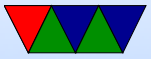
Maps to

```
    push    {lr}
    sub sp,#44

    memcpy

    add sp,#44
    pop {pc}
```





A value written to a[11] overwrites the saved link register. If you can put a pointer to a function of your choice there you can hijack the code execution, as it will be jumped to at function exit.



Mitigating Buffer Overflows

- Extra Bounds Checking / High-level Language (not C)
- Address Space Layout Randomization
- Putting lots of 0s in code (if strcpy is causing the problem)
- Running in a "sandbox"