

Embedded Systems Reverse Engineering

// WEEK 10

Conditionals in Embedded Systems:
Debugging and Hacking Static & Dynamic
Conditionals w/ SG90 Servo Motor PWM

George Mason University

RP2350 // ARM Cortex-M33

Conditionals Overview

Static vs Dynamic Decision Making

What Are Conditionals?

Structures that let programs choose different paths based on conditions

Static Conditional

Value fixed at compile time

```
int choice = 1;           // never changes
if (choice == 1)
    printf("1");          // always runs
else printf("2");          // never runs
```

Dynamic Conditional

Value changes at runtime

```
choice = getchar();
// user types a key
if (choice == '1')
    printf("1");          // maybe runs
```

if/else

Feature	Description
Condition	Any boolean expr
Values	Ranges, complex logic
Fall-through	No
Best for	2-3 conditions

switch/case

Feature	Description
Condition	Single variable
Values	Discrete only
Fall-through	Yes (no break)
Best for	Many conditions

Static Conditionals

Fixed Outcome -- Same Path Every Time

Static Code Pattern

```
int choice = 1;           // NEVER changes
while (true) {
  if (choice == 1)
    printf("1");
  else if (choice == 2)
    printf("2");           // dead code
  else
```

Execution Flow

choice == 1? — YES print "1"

NO (never taken)

choice == 2?

NO (never reached)

print "?"

Only ONE path ever executes!

Every Loop Iteration (Always the Same)

1. if(1==1) --> TRUE	2. print "1"	3. switch(1) case 1	
4. print "one"	5. servo 0deg	6. sleep 500ms	7. servo 180deg

Serial Output (Forever)

```
1
one // repeats forever
1
```

Servo Motion (Forever)

0deg --> 180deg --> 0deg

Sweeps back and forth, 500ms each

Continuous, predictable motion

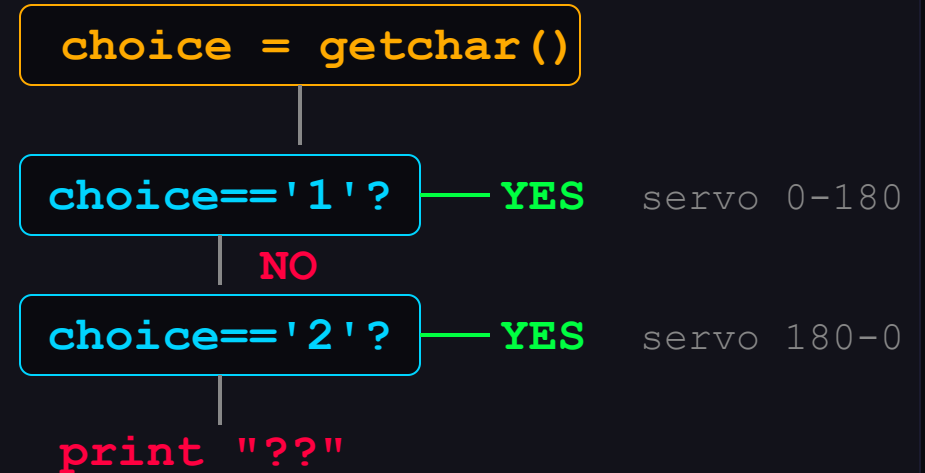
Dynamic Conditionals

Runtime Input Changes the Path

Dynamic Code Pattern

```
uint8_t choice = 0;
while (true) {
    choice = getchar();
    // waits for keyboard input
    if (choice == 0x31)
        printf("1");
    else if (choice == 0x32)
        printf("2");
}
```

Execution Flow



Each iteration can take a DIFFERENT path

getchar() Returns ASCII

'1' = 0x31 '2' = 0x32
'x' = 0x78 'y' = 0x79 Blocks until k

Input --> Behavior

Key	Output	Servo
'1'	"1" + "one"	0deg --> 180deg
'2'	"2" + "two"	180deg --> 0deg

Two Projects

0x001d_static-conditionals

choice = 1 (fixed)

0x0020_dynamic-conditionals

choice = getchar() (user input)

PWM Basics

Pulse Width Modulation for Servo Control

What is PWM?

Rapidly switching a signal ON and OFF
Ratio of on-time to off-time controls power



Servo PWM (50Hz = 20ms period)

0deg (1ms pulse): 1ms HIGH 19ms LOW

90deg (1.5ms pulse): 1.5ms HIGH 18.5ms LOW

180deg (2ms pulse): 2ms HIGH 18ms LOW

Pulse WIDTH determines angle, not duty cycle
Total period always 20ms (50Hz)

Angle to Pulse Width

Angle	Pulse	Ticks (1MHz)
0deg	1000us	1000
90deg	1500us	1500
180deg	2000us	2000

Formula

$$\text{pulse} = 1000 + (\text{angle}/180) \times 1000$$

Example for 90deg:

$$1000 + (90/180) \times 1000 \\ = 1500\text{us} = 1500 \text{ ticks}$$

PWM Timing Chain

150MHz System Clock to 50Hz Servo Signal

Clock Division

150 MHz Clock — / 150 — 1 MHz PWM — 1 tick = 1us

Step 1: $150,000,000 / 150 = 1,000,000$ Hz

Each PWM tick = exactly 1 microsecond

Step 2: Wrap at 20,000 ticks = 20ms = 50Hz Wrap value = 19,999

SG90 Servo Motor

Parameter	Value
Voltage	4.8V - 6V (use 5V)
Rotation	0deg to 180deg
Pulse Width	1000us - 2000us
Frequency	50Hz (20ms period)

Wiring to Pico 2

Pico

SG90

GPIO 6 = Signal (Orange)

VBUS 5V = VCC (Red)

GND = GND (Brown)

Add 1000uF capacitor on power!

Power Safety

NEVER use 3.3V pin for servo!

Servos draw 650mA+ (spikes to 1A)

Use VBUS (5V from USB) with 1000uF 25V capacitor

Static Source Code

0x001d_static-conditionals.c

Full Source

```
#include <stdio.h>
#include "pico/stdlib.h"
#include "servo.h"
#define SERVO_GPIO 6
int main(void) {
    stdio_init_all();
    int choice = 1;           // STATIC!
    servo_init(SERVO_GPIO);
    while (true) {
        if (choice == 1)
            printf("1\r\n");
        else if (choice == 2)
            printf("2\r\n");    // dead code
    }
}
```

switch Block

```
switch(choice) {
    case 1: puts("one"); break;
```

Servo Loop

```
servo_set_angle(0.0f);
sleep_ms(500);    // then 180
```

Dynamic Source Code

0x0020_dynamic-conditionals.c

Full Source

```
#include <stdio.h>
#include "pico/stdlib.h"
#include "servo.h"
#define SERVO_GPIO 6
int main(void) {
    stdio_init_all();
    uint8_t choice = 0;    // DYNAMIC!
    servo_init(SERVO_GPIO);
    while (true) {
        choice = getchar(); // wait for input
        if (choice == 0x31) // '1'
            printf("1\r\n");
        else if (choice == 0x32) '2'
            printf("2\r\n");
    }
}
```

switch Block (with servo control)

```
case '1': print "one", servo 0-->180, sleep 500ms
case '2': print "two", servo 180-->0, sleep 500ms
```


Branch Instructions

How Conditionals Become Assembly

ARM Branch Instructions

Instr	Meaning	Condition
b	Branch always	Always
beq	Branch if Equal	Z flag set
bne	Branch if !=	Z flag clear
bgt	Branch if >	Signed >
blt	Branch if <	Signed <

C --> Assembly

C code:

```
if (choice == 0x31)
    printf("1");
```

Assembly:

```
cmp r4, #0x31    // compare
bne skip_printf  // skip if !=
```

Conditional Branch Flow

`cmp r4, #0x31`

`beq target_addr`

`r4==0x31: JUMP`

`r4!=0x31: continue next`

cmp sets CPU flags, branch reads them

NOP (No Operation)

ARM Thumb NOP: `00 bf` 2 bytes

Wide NOP: `00 f0 00 80`

Replaces 4-byte bl instruction

Hacking Branches

Change branch target addr

Redirect program flow

NOP out instructions

Erase code silently

Hacking Conditionals

Strings, Timing, Stealth Commands

Hack 1: Change Strings

Change "1" to "2": **0x31** --> **0x32**
"one" to "fun": **6f 6e 65** --> **66 75 6e**

Hack 2: Speed Up Servo

Change sleep_ms delay:
0x1F4 (500ms) --> **0x064 (100ms)**

Hack 3: Stealth Commands

Hidden keys move servo with NO output

Patch	Original	Hacked	Purpose
Compare 1	#0x31 ('1')	#0x78 ('x')	New trigger key
Compare 2	#0x32 ('2')	#0x79 ('y')	New trigger key
puts calls	bl puts	00 bf 00 bf	NOP out prints

Hack 4: Change Angle

180.0f --> 30.0f:
00 00 34 43 --> **00 00 f0 41**

Stealth Result

'1','2': normal output + servo
'x','y': **NO output, servo moves**

Workflow

Patch bytes in Ghidra --> export .bin --> convert to UF2 --> flash to Pico

PWM & Servo Hacking

Conditionals, PWM, Servo, and Hacking

Static vs Dynamic

Static `choice = 1 (fixed)`
Same path every iteration
Compiler may optimize

Dynamic `choice = getchar()`
Different paths at runtime

PWM for Servos

`150MHz / 150 = 1MHz tick`
`Wrap 20000 = 50Hz (20ms)`
`0deg=1000us 90deg=1500us 180deg=2000us`
`pulse = 1000 + (angle/180) x 1000`

Branch Instructions

`cmp r4, #0x31` Compare
`beq target` Jump if equal
`bne target` Jump if not equal
NOP = `00 bf` (erase code)

Key Values

`0x10000234` `main()`
`0x40070000` `UART0`
`0x1F4` `500 (sleep_ms)`
`0x43340000` `180.0f IEEE-754`

4 Hack Types Applied

String	<code>"one"-->"fun"</code>	Timing	<code>500ms-->100ms</code>
Stealth	<code>NOP out prints</code>	Angle	<code>180.0f-->30.0f</code>

Projects

`0x001d_static-conditionals`
`0x0020_dynamic-conditionals`

IEEE-754 Angles

`0.0f=00000000 90.0f=42b40000`
`180.0f=43340000 30.0f=41f00000`