06/07: Embedded architectures & MCU datasheets

Review

- Sensors and actuators (I/O devices) can be analog or digital
- MCUs can read from/write to I/O devices
  - GPIO pins (for digital signals and PWM)
  - DACs, ADCs (for analog signals)
  - This enables us to use software to interact with the physical world

General purpose computing and I/O

When you run code on a computer:

- How is the program able to read input (user/keyboard input or files)?
- How is the program able to create output (write to files/draw pixels on a screen)?

Operating system provides interfaces that translate commands into appropriate hardware actions

MCUs are varied

Wikipedia list

Variations in:

- Word size
- Memory
- I/O and peripherals

Without an OS, it is the programmer's role to understand the specifics of the MCU hardware

Architecture

The organization and design of a computer

Defines the SW/HW interface

- Instruction Set Architecture (ISA): *machine code, word sizes, memory addresses, data formats, registers*
- Microarchitecture (implementation of ISA): CPU internals, memory hierarchy
- Systems design: all other HW support

How software you write becomes code running on an CPU

Code you write

Assembly Code

Machine code

Program

```
int N = 12;
int fibo = 0;

void setup() {
  int f_prev = 1;
  int f = 1;

  int i = 0;

  while (i < N) {
    int f_next = f + f_prev;
    f_prev = f;
    f = f_next;
    i += 1;
  }
  fibo = f;
}

void loop() {
  Serial.println(fibo);
  delay(100);
}
```

Assembly

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

Memory address of instruction

Instruction in machine code (hex)

Assembly instructions

Types of data

```
ldr r3, [pc, #28]    ; (211c <setup+0x20>)
push    {r4, lr}
ldr r4, [r3, #0]
movs    r3, #1
movs    r2, #0
movs    r1, r3
cmp r4, r2
ble.n   2116 <setup+0x1a>
adds    r0, r1, r3
adds    r2, #1
movs    r1, r3
movs    r3, r0
b.n 2108 <setup+0xc>
ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
str r3, [r2, #0]
pop {r4, pc}
```

registers

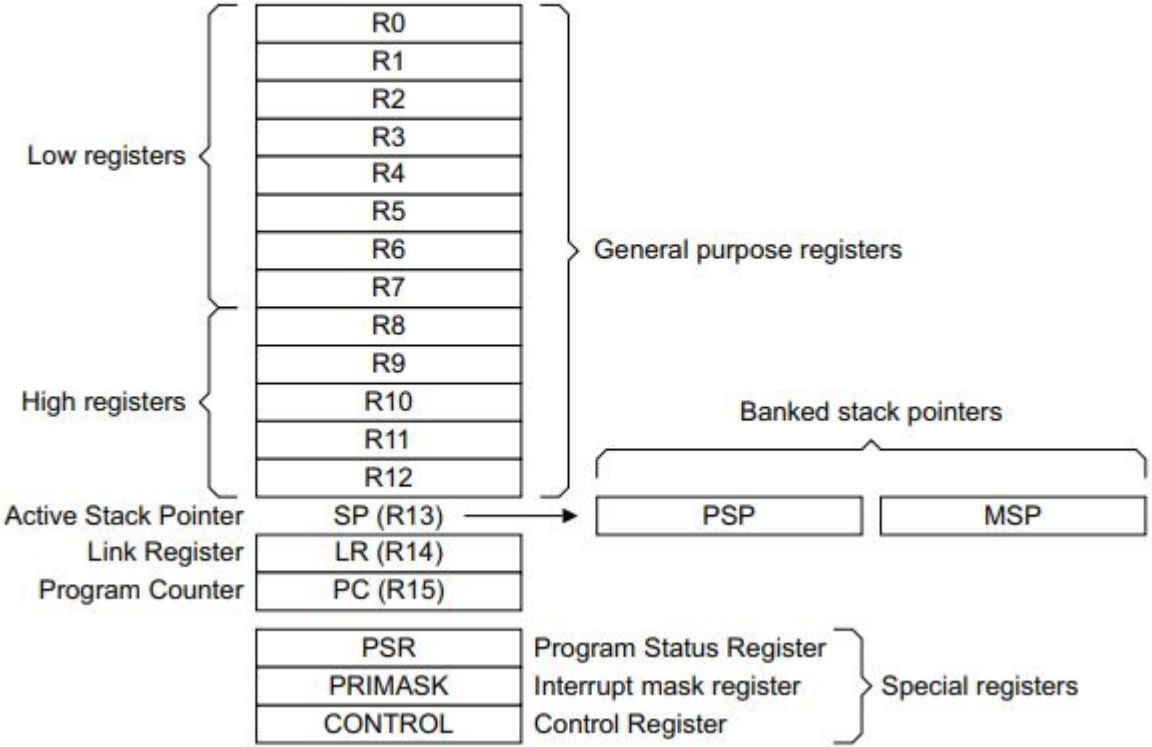
immediates

memory addresses
(raw or computed from
registers/immediates)

Registers

- Small pieces of fast memory
- Usually 8-, 16-, 32- or 64-bits
- Many purposes on CPUs and MCUs:
  - Storing temporary data for execution
  - Addressing memory
  - Configuring peripherals (Lab 3)

The processor core registers are:



Types of operations

```
ldr r3, [pc, #28]    ; (211c <setup+0x20>)
push    {r4, lr}
ldr r4, [r3, #0]
movs    r3, #1
movs    r2, #0
movs    r1, r3
cmp r4, r2
ble.n   2116 <setup+0x1a>
adds    r0, r1, r3
adds    r2, #1
movs    r1, r3
movs    r3, r0
b.n 2108 <setup+0xc>
ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
str r3, [r2, #0]
pop {r4, pc}
```

register operations

stack operations

memory loads/stores

control logic

Stack

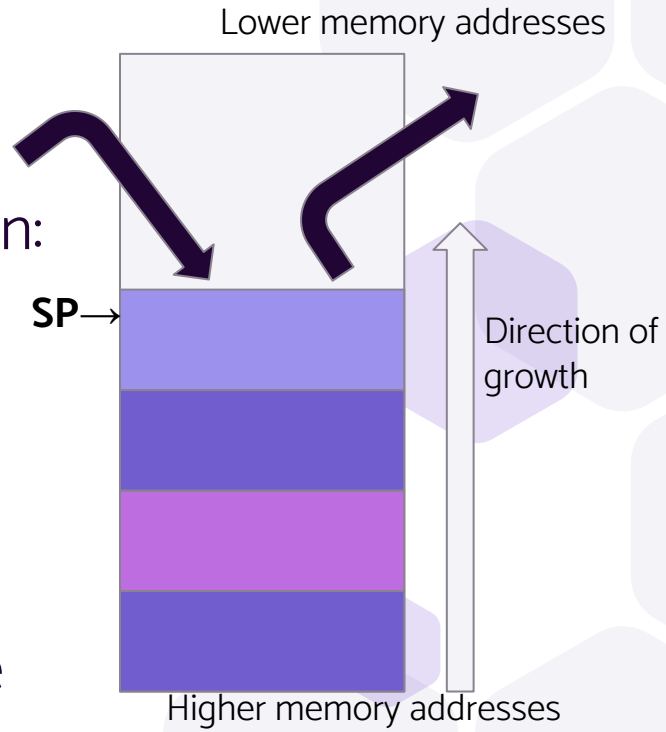
LIFO (last-in, first-out) data structure

Keeps track of information for execution:

- Local variables
- Return pointers

Grows “downward”

Stack Pointer (SP) points to latest value



What a processor needs to do:

Fetch an instruction from memory

Decode the instruction

Execute arithmetic and logical operations

Load/store values in **Memory**

Write back values to registers

Different CPUs do these in different orders/groupings and in different ways

Cortex-M0+

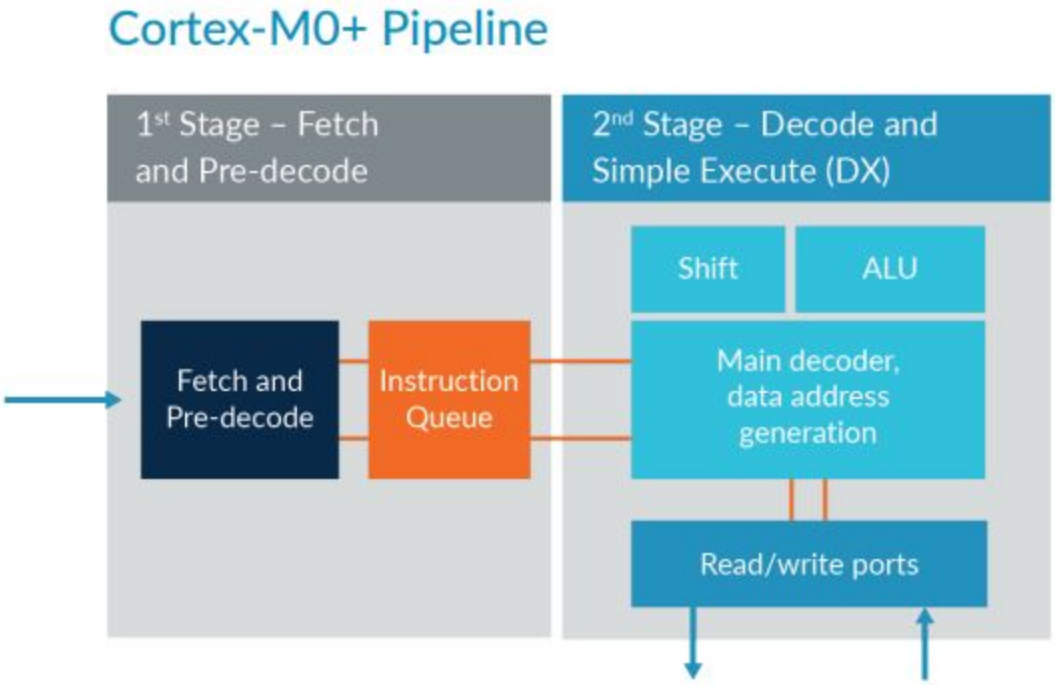


Image source

Memory

Information stored in memory:

- Code
- Program data
- Stack
- Heap (dynamically allocated data)
- Register file

```
Disassembly of section .data:

20000000 <N>:
20000000:       0000000c
```

Every location in memory has an **address**

Types of memory

Volatile - Gets erased when power gets turned off

- RAM (DRAM, SRAM)

Non-volatile - Persists when power gets turned off

- Flash
- ROM (sometimes rewritable, like EEPROM)

Specifications

x-bit processor:

- Data registers, data buses, *words* are that size

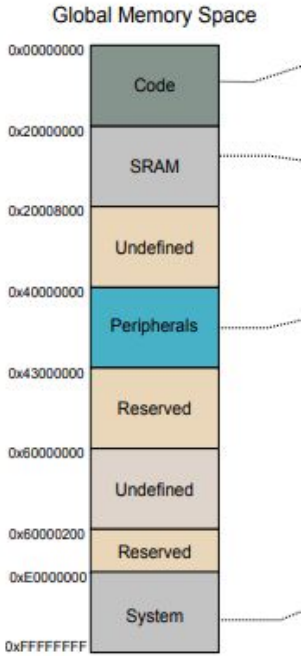
memory address may not be that size

- Common for 8-bit CPUs to have 16-bit addresses (**why?**)
- **What are the implications for atomicity?**

Harvard Architecture - code has separate memory space from data (common in MCUs)

- vs. Von Neumann - shared memory space (SAM D21 is Von Neumann)

Memory layout of SAMD 21 chip



How information gets onto an MCU

Bootloader

- Firmware on the board that can interface with the computer
- Copies memory on upload

Hardware programmer

- Special piece of hardware that connects to pins directly and transfers using a protocol

Peripherals

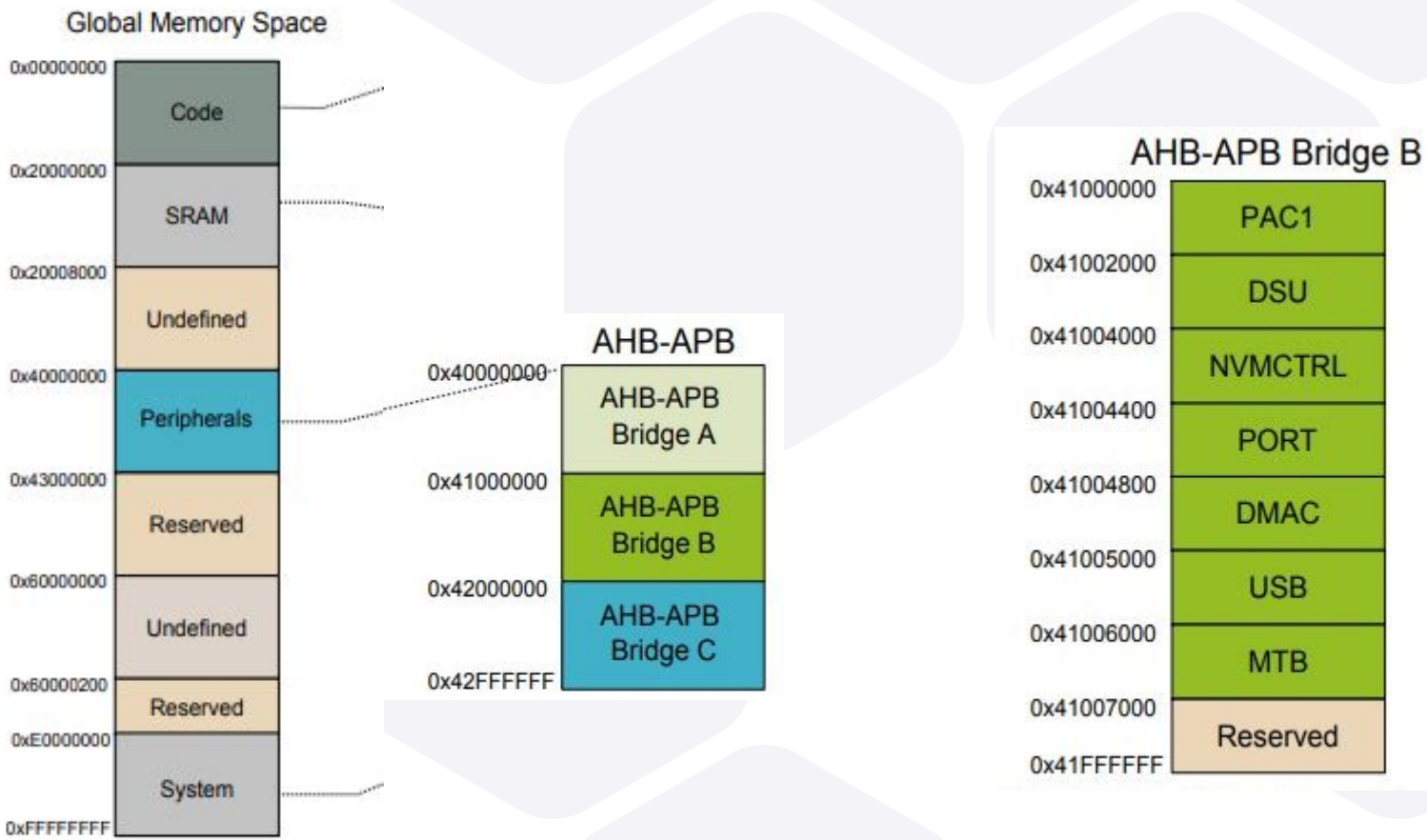
Timers, ADCs, GPIO, etc

Controlled by special registers (**different** from CPU registers!)

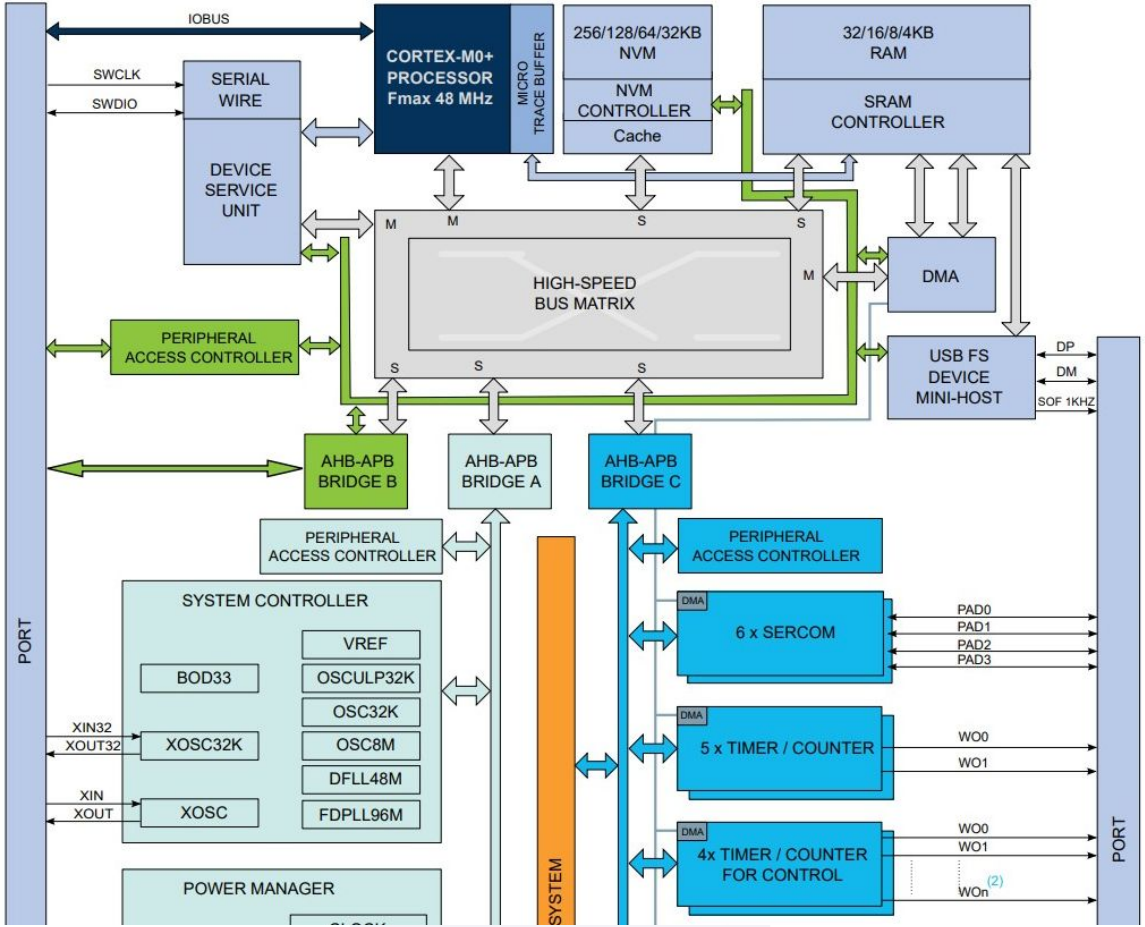
- You will see this in lab!

"Memory-mapped": from CPU perspective, just like writing to any other memory address

- From MCU perspective, need controller hardware to configure/send data to the right place

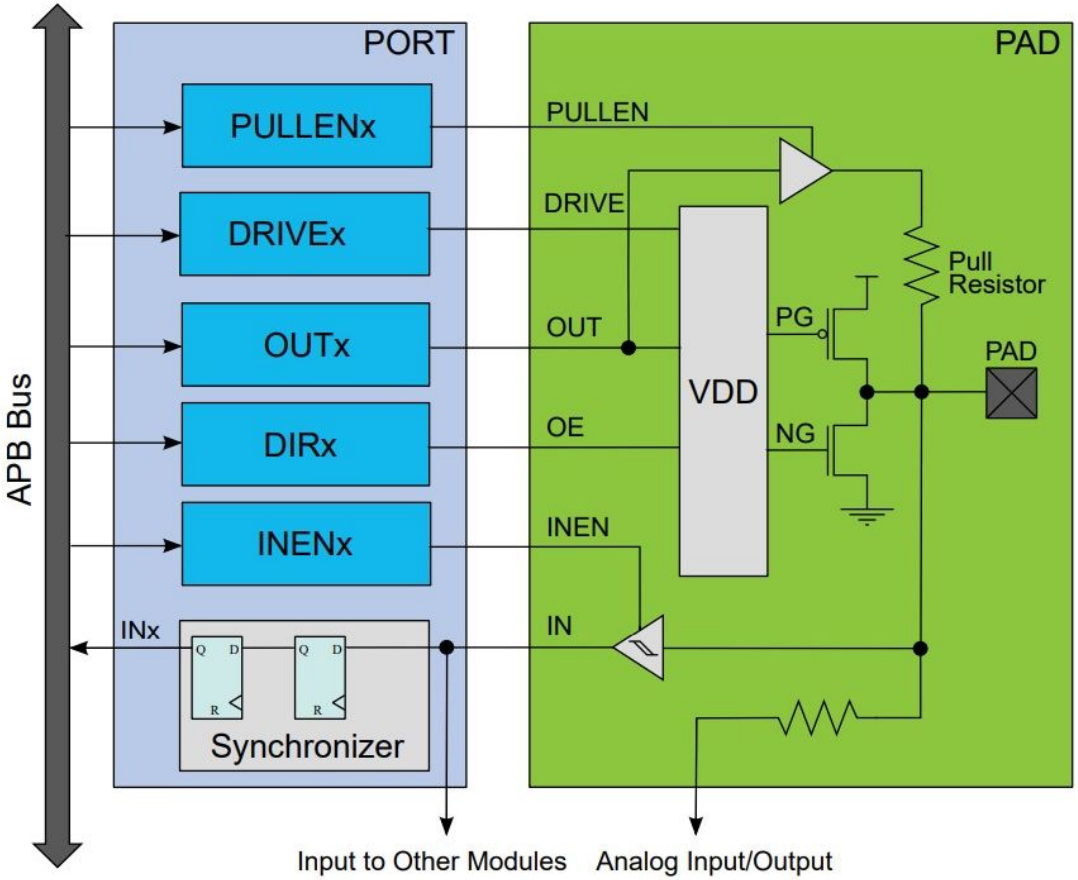

Global Memory Space
0x00000000
Code
0x20000000
SRAM
0x20008000
Undefined
0x40000000
Peripherals
0x43000000
Reserved
0x60000000
Undefined
0x60000200
Reserved
0xE0000000
System
0xFFFFFFFF
AHB-APB
0x40000000
AHB-APB Bridge A
0x41000000
AHB-APB Bridge B
0x42000000
AHB-APB Bridge C
0x42FFFFFF
AHB-APB Bridge B
0x41000000
PAC1
0x41002000
DSU
0x41004000
NVMCTRL
0x41004400
PORT
0x41004800
DMAC
0x41005000
USB
0x41006000
MTB
0x41007000
Reserved
0x41FFFFFF

4. Block Diagram



23.6 Functional Description

Figure 23-2. Overview of the PORT



MCU datasheet example

Configure and write to DAC

Resources used in this presentation

ARM Cortex M0+ devices generic user guide

ARMv6-M Architecture reference manual

Supplement: execution of assembly

A2.3.1 ARM core registers

There are thirteen general-purpose 32-bit registers, R0-R12, and an additional three 32-bit registers that have special names and usage models:

SP Stack Pointer, used a pointer to the active stack. For usage restrictions see *Use of 0b1101 as a register specifier* on page A5-83. This is preset to the top of the Main stack on reset. See *The SP registers* on page B1-211 for more information. SP is sometimes referred to as R13.

LR Link Register stores the Return Link. This is a value that relates to the return address from a subroutine that is entered using a Branch with Link instruction. The LR register is also updated on exception entry, see *Exception entry behavior* on page B1-224. LR is sometimes referred to as R14.

Note

LR can be used for other purposes when it is not required to support a return from a subroutine.

PC Program Counter, see *Use of 0b1111 as a register specifier* on page A5-82 for more information. The PC is loaded with the Reset handler start address on reset. PC is sometimes referred to as R15.

Cortex M0+ stack operations

push *reglist* - push the registers in *reglist* onto the stack (highest value registers pushed first), decrements stack pointer

pop *reglist* - pop the values on the stack into the registers in *reglist* (lowest value registers popped first)

if SP is in *reglist*, branch to where SP is pointing after pop

Loads and stores

An instruction like `ldr r1 [r2, #8]` means:

- Add 8 to the value in register r2
- Interpret the result as a memory address
- Take the value stored at that memory address and put it in r1

(Similar with `str`, which is for storing values in registers to memory addresses)

mem. address 20fc + 28 = 211c

PC

```
00000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

value at that memory address

R0	
R1	
R2	
R3	2000 0000
R4	
R5	
R6	
R7	

previous stack

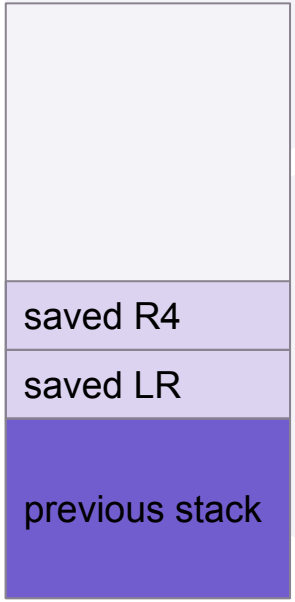
LR	

LR (R14) pushed first because 14 > 4

PC

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc,
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```



R0	
R1	
R2	
R3	2000 0000
R4	
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc,
    20fe:   b510        push    {r4,
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

mem. address 2000 0000

R0	
R1	
R2	
R3	2000 0000
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

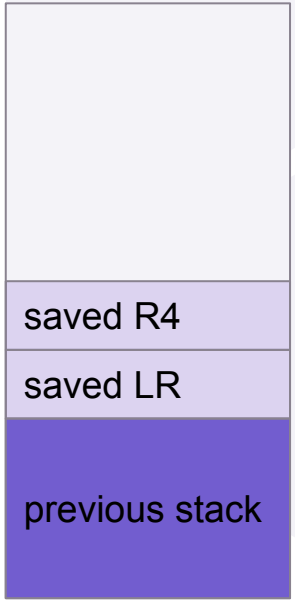
saved R4
saved LR
previous stack

LR	

PC

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```



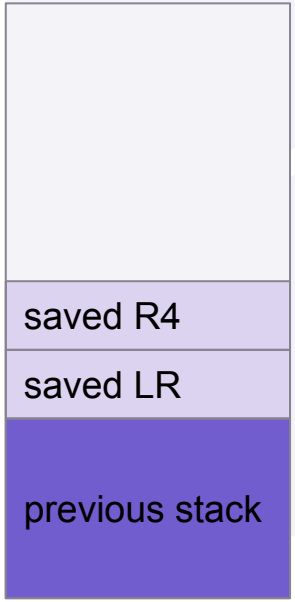
R0	
R1	
R2	
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

PC

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```



R0	
R1	
R2	0
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	
R1	1
R2	0
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r
    2104:   2200        movs    r
    2106:   0019        movs    r
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

Compute R4-R2 and set comparison flags

saved R4	
saved LR	
previous stack	

R0	
R1	1
R2	0
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

if r4 <= r2, jump to instruction at 2116 (otherwise, keep going)

saved R4
saved LR
previous stack

R0	
R1	1
R2	0
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	2
R1	1
R2	0
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	2
R1	1
R2	1
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	2
R1	1
R2	1
R3	1
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	2
R1	1
R2	1
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

jump to instruction at address 2108

R0	2
R1	1
R2	1
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

saved R4

saved LR

previous stack

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs
    2104:   2200        movs
    2106:   0019        movs
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

Compute R4-R2 and set comparison flags

saved R4	
saved LR	
previous stack	

R0	2
R1	1
R2	1
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]    ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]     ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]    ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]    ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100     ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop          ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

if r4 <= r2, jump to instruction at 2116 (otherwise, keep going)

R0	2
R1	1
R2	1
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

saved R4
saved LR
previous stack

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	3
R1	1
R2	1
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	3
R1	1
R2	2
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	3
R1	2
R2	2
R3	2
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

saved R4
saved LR
previous stack

R0	3
R1	2
R2	2
R3	3
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

jump to instruction at address 2108

R0	3
R1	2
R2	2
R3	3
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

saved R4

saved LR

previous stack

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setup+0xc>
    2116:   4a02        ldr r2, [pc, #8]    ;
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ;
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7printlnEii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

jum

Essentially, this code is doing a loop, running the following computation:
r0 = r1 + r3
r2 = r2 + 1
r1 = r3
r3 = r0

And looping while r2 < r4 (so, r2 is a counter and r0/r1/r3 are used to compute the next fibonacci number). Going back to our original code, this suggests that the value at r4 (memory location 2000 0000) is 12 (N), and we'll run 12 iterations of this loop

R0	3
	m. value at 0 0000)
LR	

previous stack

AFTER THE LOOP COMPLETES
(ble.n 2116 at instruction 210a is executed)

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r0
    2114:   e7f8        b.n 2108 <setu
    2116:   4a02        ldr r2, [pc, #8]    ; (2120 <setup+0x24>)
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

address 2116 + 8 = 2120

saved R4	
saved LR	
previous stack	

R0	(final value of R0)
R1	(final value of R1)
R2	2000 00bc
R3	(final value of R3)
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

LR	

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r2, #1
    2110:   0019        movs    r1, r3
    2112:   0003        movs    r3, r
    2114:   e7f8        b.n 2108 <set
    2116:   4a02        ldr r2, [pc,
    2118:   6013        str r3, [r2, #0]
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

PC

store final R3 value at address 2000 00bc

R0	(final value of R0)
R1	(final value of R1)
R2	2000 00bc
R3	(final value of R3)
R4	(mem. value at 2000 0000)
R5	
R6	
R7	

saved R4
saved LR
previous stack

LR	

PC

saved LR

```
000020fc <setup>:
    20fc:   4b07        ldr r3, [pc, #28]   ; (211c <setup+0x20>)
    20fe:   b510        push    {r4, lr}
    2100:   681c        ldr r4, [r3, #0]
    2102:   2301        movs    r3, #1
    2104:   2200        movs    r2, #0
    2106:   0019        movs    r1, r3
    2108:   4294        cmp r4, r2
    210a:   dd04        ble.n   2116 <setup+0x1a>
    210c:   18c8        adds    r0, r1, r3
    210e:   3201        adds    r
    2110:   0019        movs    r
    2112:   0003        movs    r
    2114:   e7f8        b.n 2108
    2116:   4a02        ldr r2, [
    2118:   6013        str r3, [r2, #0
    211a:   bd10        pop {r4, pc}
    211c:   20000000    .word   0x20000000
    2120:   200000bc    .word   0x200000bc

00002124 <loop>:
    2124:   b510        push    {r4, lr}
    2126:   4b05        ldr r3, [pc, #20]   ; (213c <loop+0x18>)
    2128:   220a        movs    r2, #10
    212a:   6819        ldr r1, [r3, #0]
    212c:   4804        ldr r0, [pc, #16]   ; (2140 <loop+0x1c>)
    212e:   f002 f8d6   bl  42de <_ZN7arduino5Print7println Eii>
    2132:   2064        movs    r0, #100    ; 0x64
    2134:   f000 f8c2   bl  22bc <delay>
    2138:   bd10        pop {r4, pc}
    213a:   46c0        nop         ; (mov r8, r8)
    213c:   200000bc    .word   0x200000bc
    2140:   200001a4    .word   0x200001a4
```

popping into PC branches to that address (effectively returning from a subroutine)

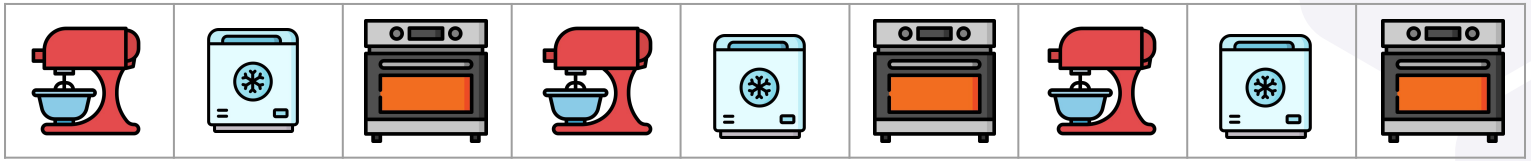
previous stack

R0	(final value of R0)
R1	(final value of R1)
R2	2000 00bc
R3	(final value of R3)
R4	**saved** R4
R5	
R6	
R7	

LR	

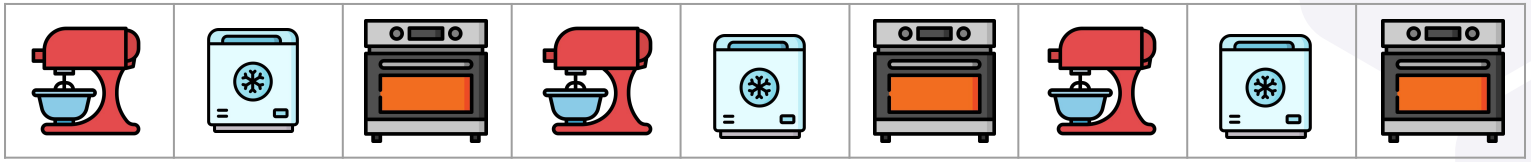
Pipelining slides (not used this semester)

Sequential execution

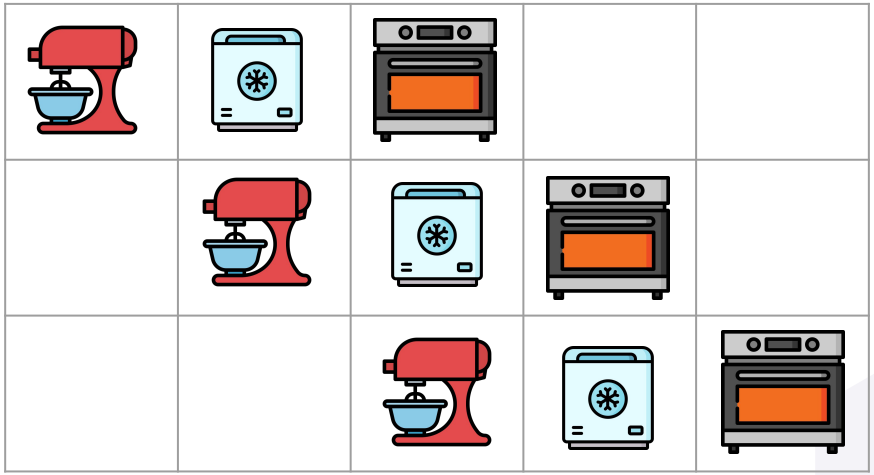


images: Flaticon.com

Pipelining

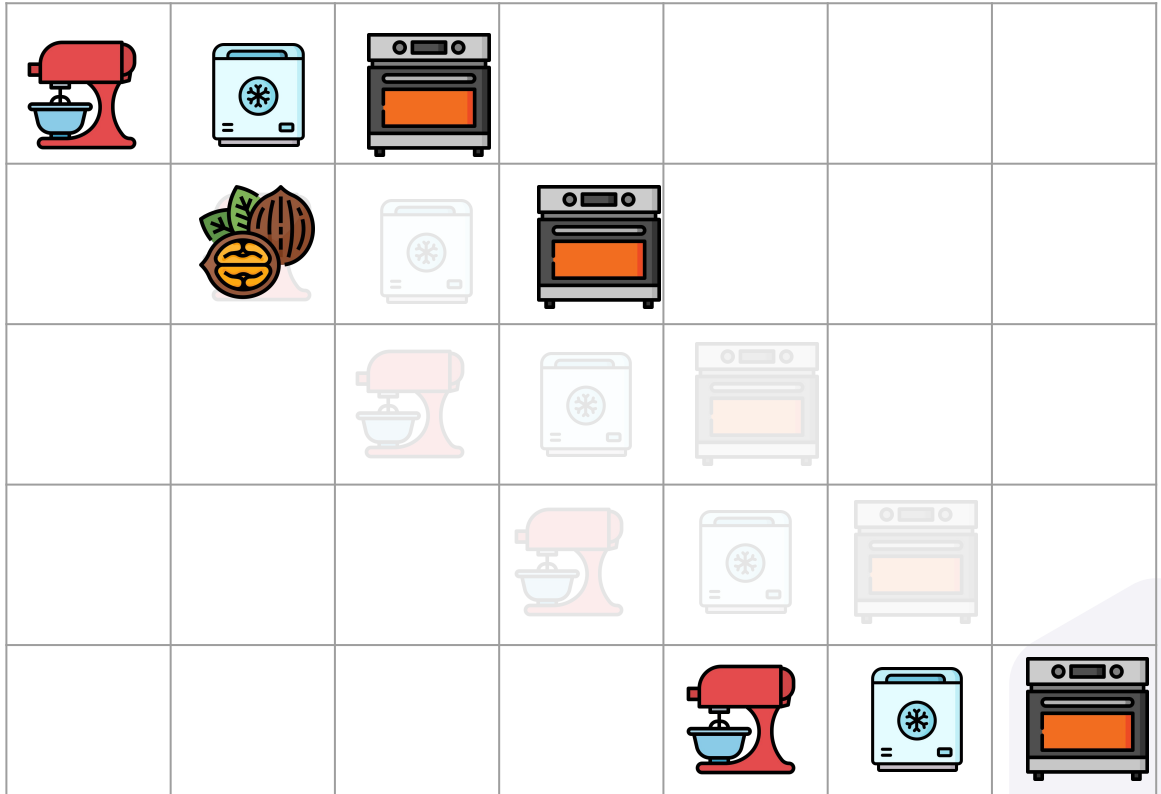


vs



images: Flaticon.com

Pipeline hazards



images: Flaticon.com

Avoiding (some) hazards using compilation

```
while (i < N) {
  int f_next = f + f_prev;
  f_prev = f;
  f = f_next;
  i += 1;
}
```

```
cmp r4, r2
ble.n   2116 <setup+0x1a>
adds    r0, r1, r3
adds    r2, #1
movs    r1, r3
movs    r3, r0
b.n 2108 <setup+0xc>
```

Can you summarize the tradeoff between deep and shallow pipelines and predict which kind MCUs are more likely to have?