A/D & D/A II

ECE 3710

Yesterday I saw a chicken crossing the road. I asked it why. It told me it was none of my business.

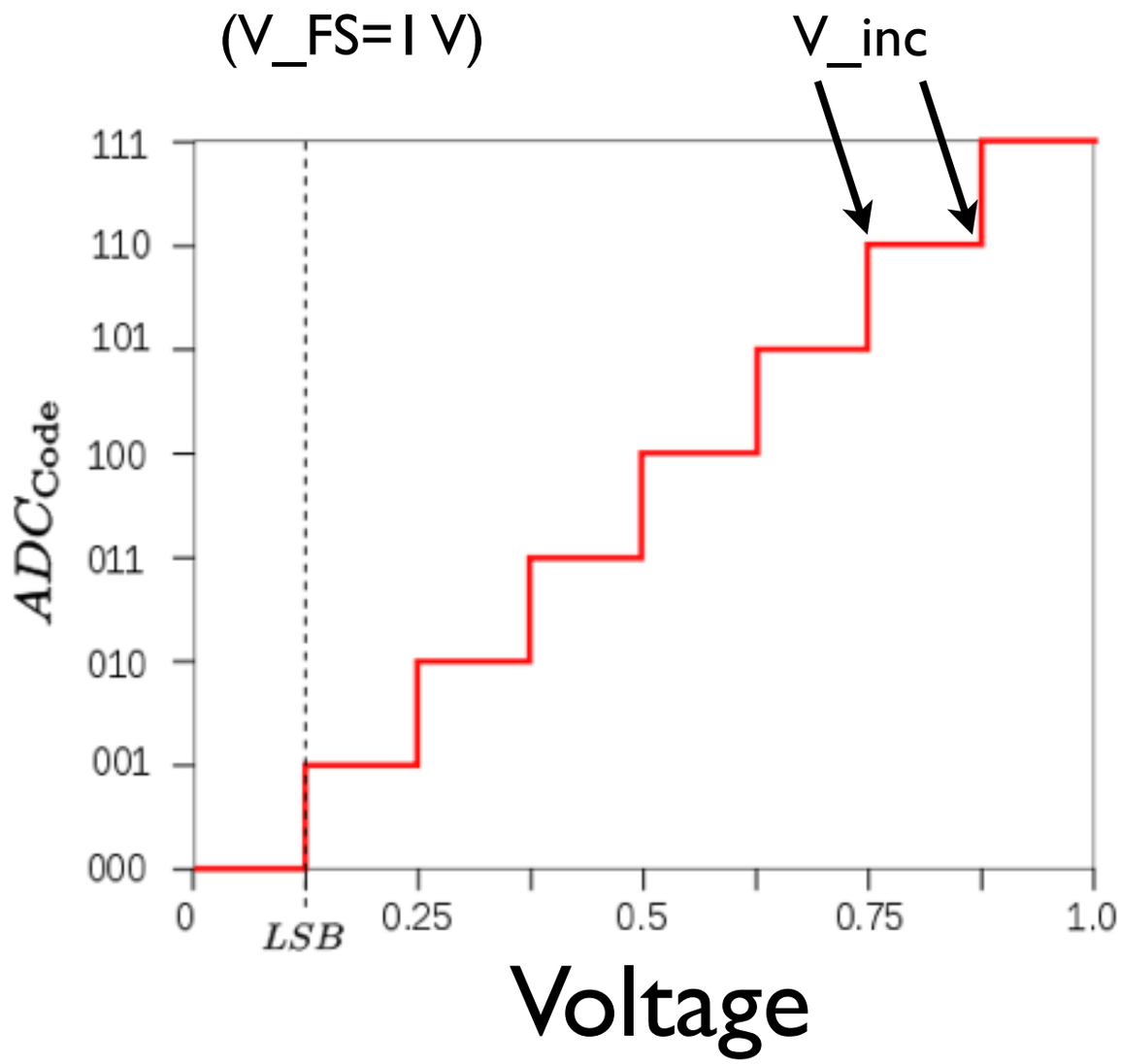
- Steven Wright

ADC: voltage ⟶ code

DAC: code ⟶ voltage

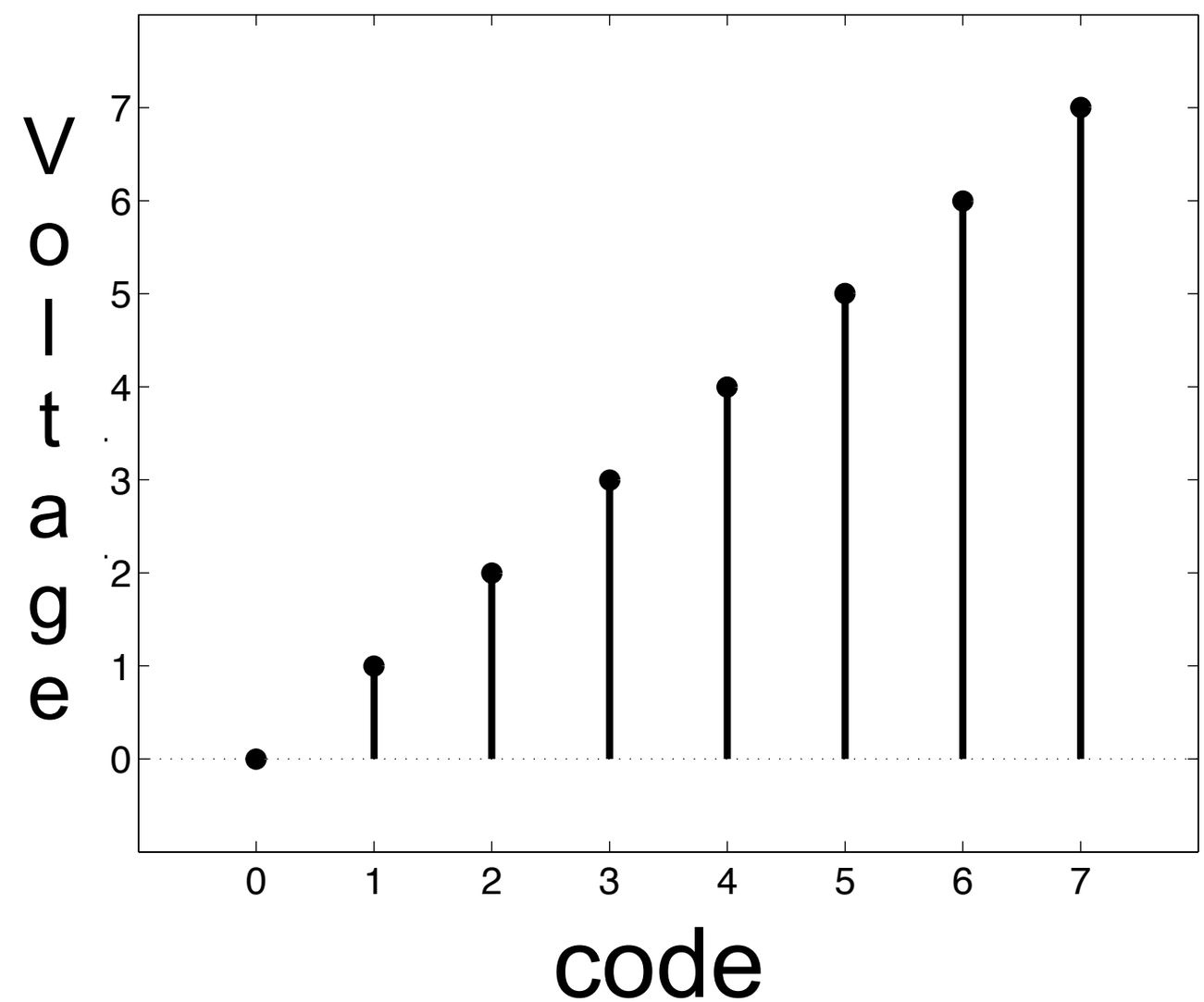
3-bit ADC:

(V_FS=1 V)

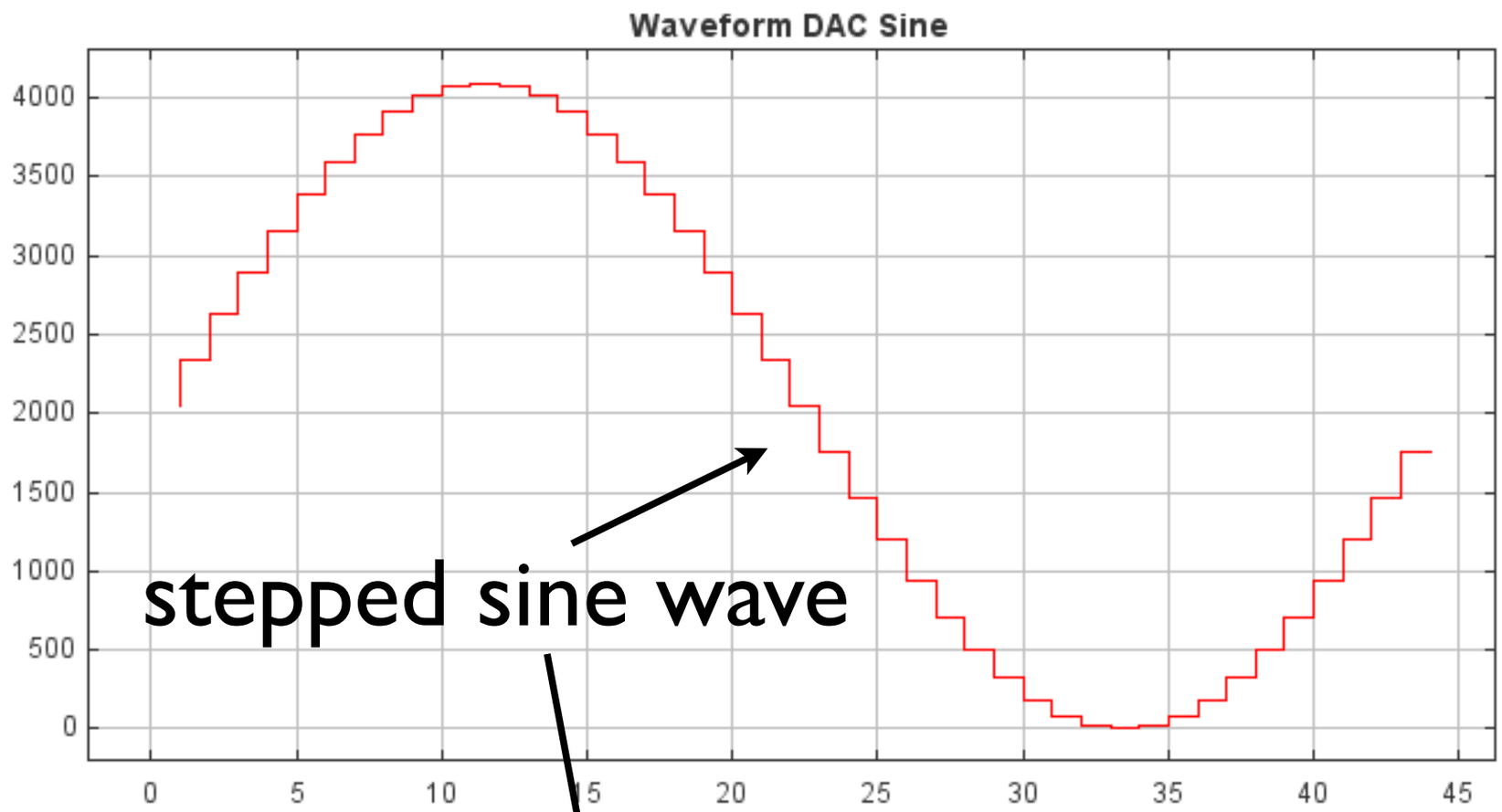


3-bit DAC:

(V_FS=7 V)



more interesting waveforms?



stepped sine wave

can only update DAC so often:

1. settling time
2. uC speed

sine wave generation: table method

decisions:

1. how many points of wave (angles) will we reproduce
2. amplitude
3. offset
4. voltage increment

limited by V_max and V_min

ceiling func.

$$\text{code} = \lceil V_{inc}^{-1}(B + A \times \sin(x)) \rceil$$

for arbitrary angle of waveform

degrees

ex: sine wave generation w/10 entries

(use full range of DAC)

angles = 0,40,80,120,160,200,240,280,320,360

B = 1.65 V

A = 1.65 V

simulator is 0--3.3 V

and 10-bits

V_inc = 3.3/(2^10);

for each angle:

$$\text{code} = \lceil V_{inc}^{-1}(B + A \times \sin(x)) \rceil$$

codes = 512, 842, 1017, 956, 688, 337, 69, 8, 183, 512

getSineCode.m

ex: sine wave generation w/10 entries

```
int main(void)
{
  unsigned int i;
  unsigned int delay;

  DACInit();

   while(1)
      for(i=0;i<10;i++)
         {
            DAC_CR = (SIN_10[i]<<6);
            for(delay=0;delay<1024;delay++);
         }
}
```

to make sine wave more apparent on logic analyser

Q:

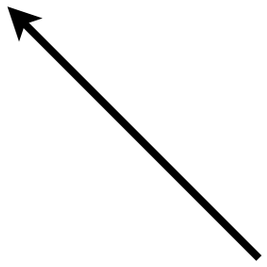
how to specify frequency of sine wave?

Q:

how to specify frequency of sine wave?

A:

go through table at 1/frequency



it should take this long to go through table

how to produce sine wave of specified frequency

let:

1. n be number of table entries
2. f be the desired frequency

do:

1. find period $\rightarrow$ $T = \frac{1}{f}$
2. output each table entry for $\rightarrow$ $\frac{T}{n}$

ex: 4 kHz sine wave w/ 25-entry table

freq. independent

codes (B=1.65, A=1.65):
512,645,768,875,956,1007,1024,1007,956,875,768,
645,512,380,256,150,69,18,0,18,69,150,256,380,
512

$$f = 4 \text{ kHz} \rightarrow T = \frac{1}{4e+3} = 250 \ \mu\text{s}$$

$$\frac{250 \ \mu\text{s}}{25} = 10 \ \mu\text{s}$$

setup timer to overflow this often

ex: 4 kHz sine wave w/ 25-entry table

(SysClk = 12 MHz)

procedure:

1. configure SysTick to expire every 10 us
2. SysTick ISR should update DAC register

(keep track of which code to place there)

ex: 10 us delay with SysClk = 12 MHz

$$(INITIAL + 1) \times \frac{1}{12\text{e}6}$$

$$n \times 0.083\mu\text{s} = 10\mu\text{s} \rightarrow n = 121$$

$$INITIAL + 1 = 121 \rightarrow INITIAL = 120$$

0x78

ex: 4 kHz sine wave w/ 25-entry table

(SysClk = 12 MHz)

```
  AREA  DATA, ALIGN=2
I DCD 0 ;where we are in entry table

  AREA     |.text|, CODE, READONLY, ALIGN=2
SIN_25 DCD 512,645,768,875,956,...,512
```

allocates one word for each entry

pseudo-code:

```
for(I=0;I<25;I++)
 DAC_CR = SIN_25[I]
```

ex: 4 kHz sine wave w/ 25-entry table

(SysClk = 12 MHz)

```
SysTick_Handler
  ; procedure: get current entry in table,
  ;  put in dac reg, increment position in table
  ; 1. get current table entry
  ldr R1,=I ;addr of counter
  ldr R0,[R1] ;get location in table
  ldr R3,=SIN_25 ;location of table
  ldr R2,[R3,R0] ;get current entry in table (SIN_25[I])
  ; 2. update dac reg
  ldr R3,=DAC_CR
  str R2,[R3]
  ; 3. update position in table
  cmp R0,#0x60 ;24*4=96=0x60
  ITE LT
  addlt R0,#4 ;go to next entry (DCD allocates a word)
  movge R0,#0 ;go back to beginning of table
  str R0,[R1] ;update location in table
  bx LR
```

note: can't produce every frequency

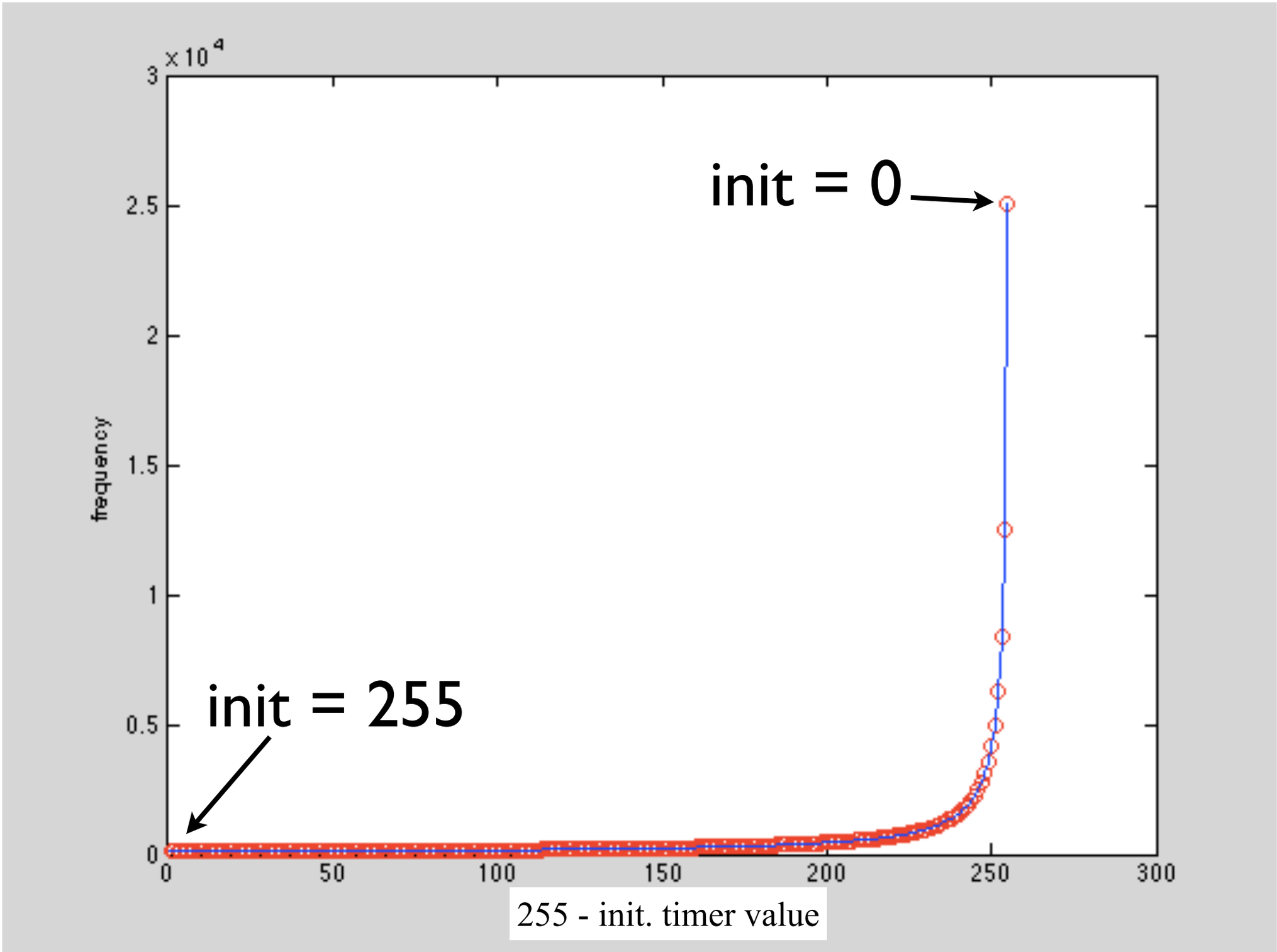
let:

1. `SysClk` = 1 MHz
2. 8-bit timer → *timers* = 255:0
3. 40 entry table

possible freqs = ceil(1./(40*12e-6*((2^8-1) + 1 - *timers*)))

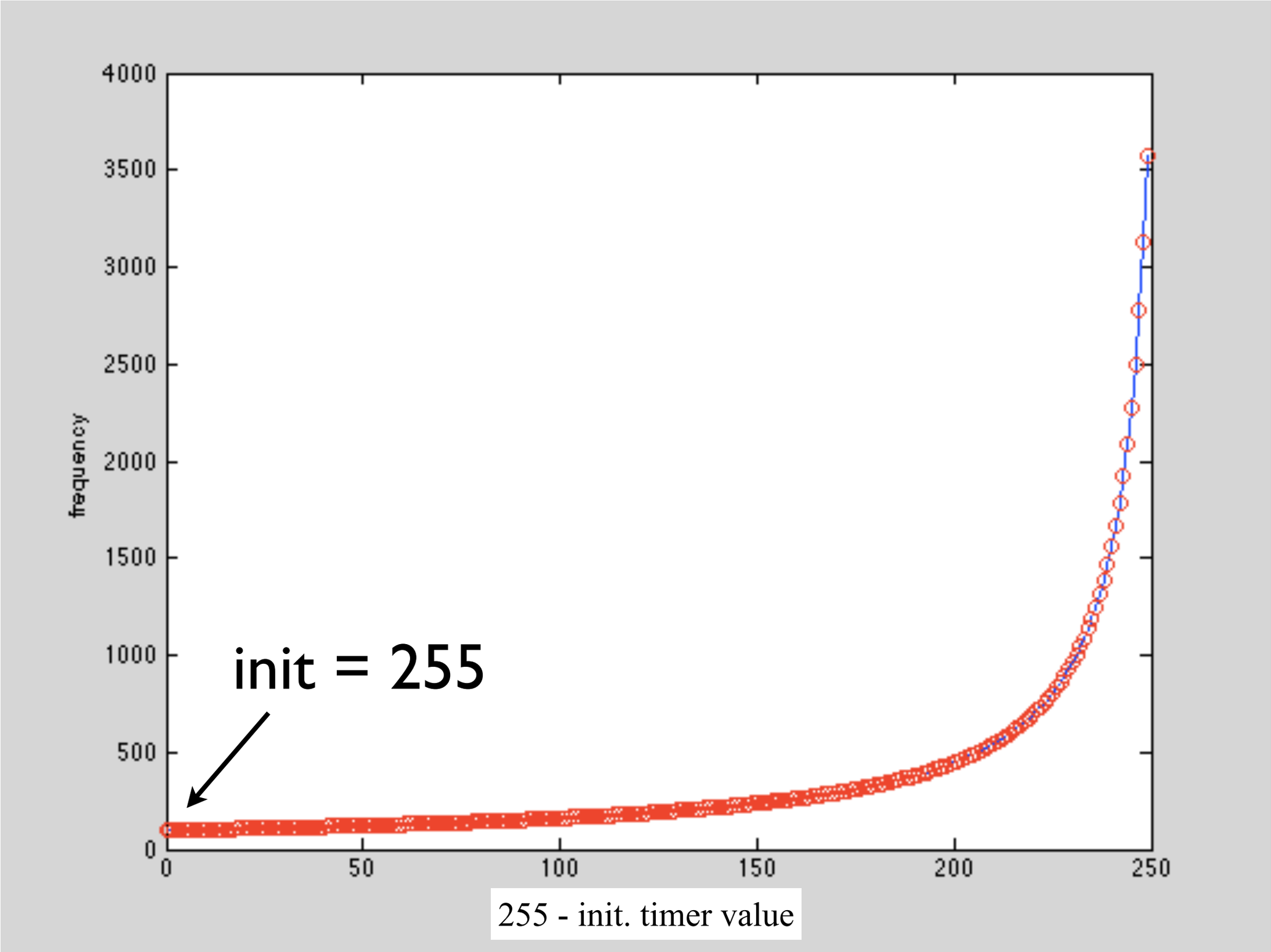
lesson: choose `SysClk` carefully

note: can't produce every frequency



lesson: choose `SysClk` carefully

note: can't produce every frequency

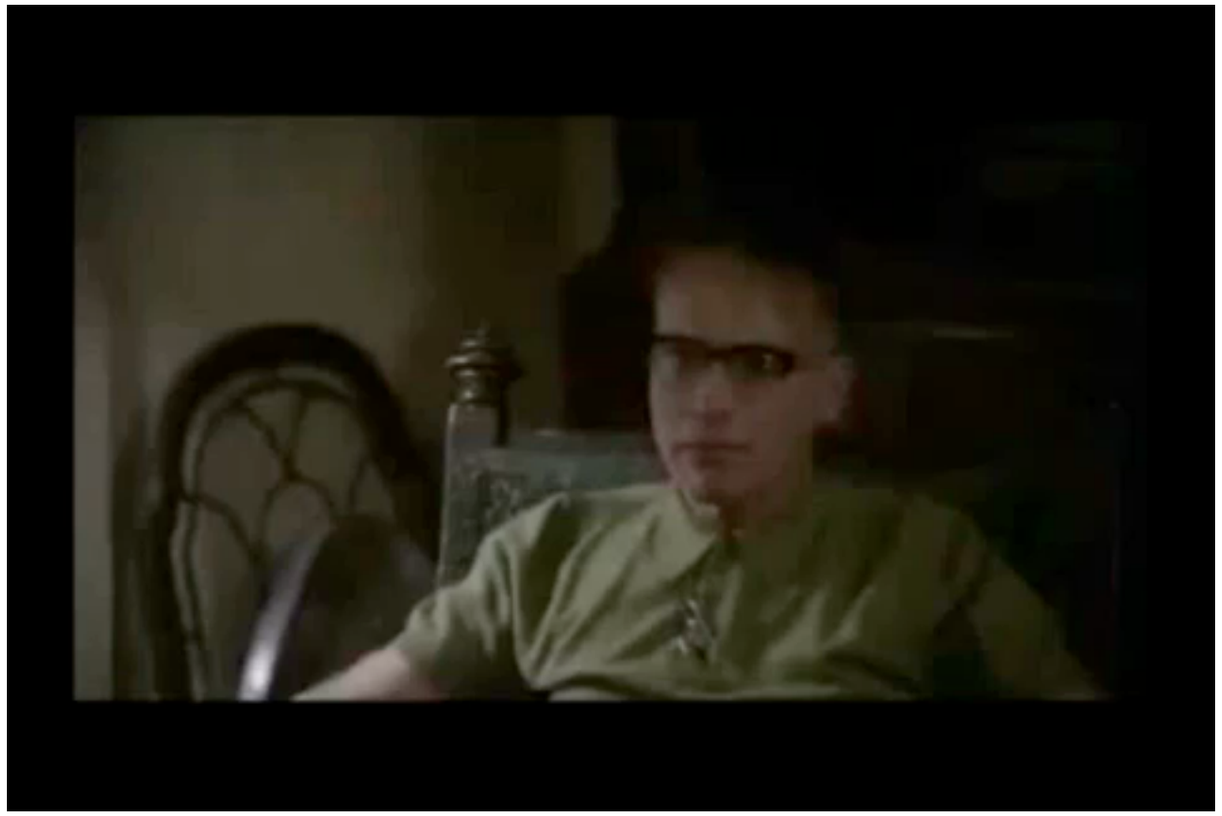


lesson: choose `SysClk` carefully

your psyche after the exam...

...and then after receiving grade

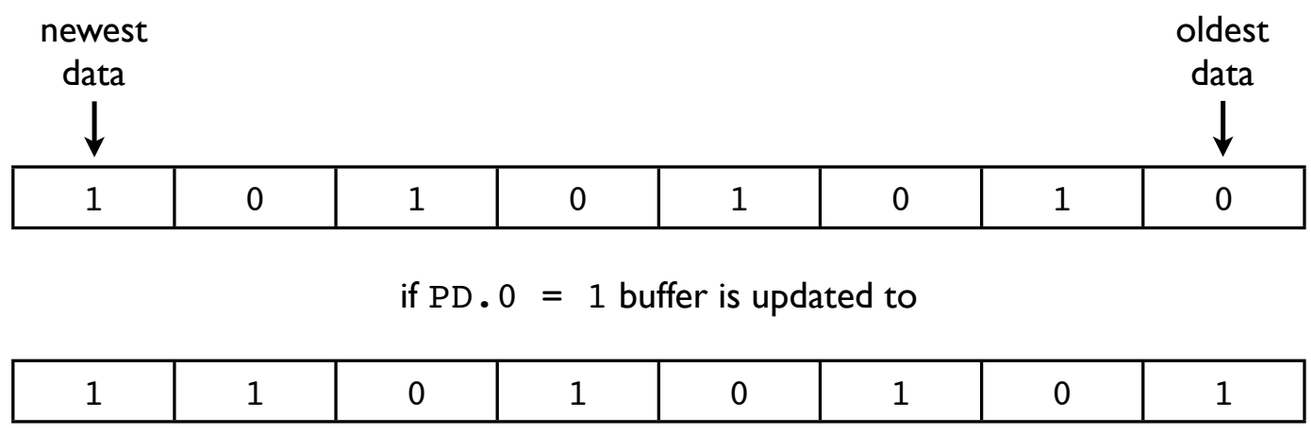
remember...



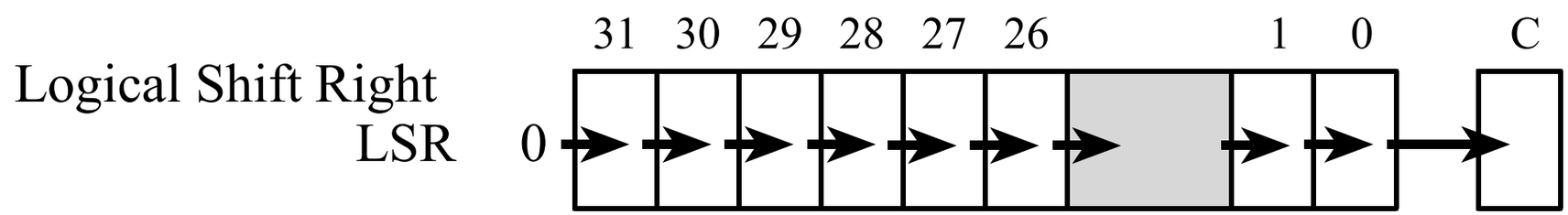
He romale, no passarán
He chavale, non rien de rien
Hey, venceremos,
Все равно мы победим!
Last one, last one goes the hope
- Gogol Bordello

this class: understanding will come
(even if it's at the end)

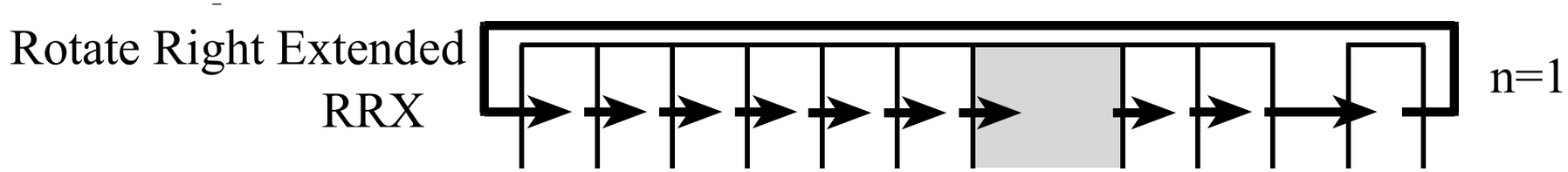
exam 1.



1. read value and shift to C of APSR:



2. shift C into buffer:



exam 2.

pass by reference: *memory location* of var. in `Rx` or on stack

pass by value: var. in `Rx` or on stack

what if n = 0?

Homework

exam 3.

T = 1/1e3 = 1 ms

Ton = 1/4*1e-3 = 250 us

close enough

n*1/6e6 = 250 us => n = 1500

notes:

1. tell, don't show, configuration
2. assume BB addr. of PD0 via label/register
3. use SysTick

exam 4.

slide 5

```
00000008: EB050706 add R7,R5,R6
```

```
0x00000134  EB000103   ADD    r0,r1,r3
0x00000138  E004       B      fun (0x00000143)
0x0000013A  BF00       NOP
0x0000013C  BF00       NOP
0x0000013E  BF00       NOP
0x00000140  BF00       NOP
fun
0x00000143  F1000010   ADD    r0,r0,#10
```

makes for positive branch

```
00000000 6808        label1 LDR R0,[R
00000002 F101 0104          ADD R1, #
00000006 E7FB               B  label1
```

offset = 0x0 - 0x6 - 4 ← PC will be one/two instructions ahead b/c of pipeline
=-10
=0b111111110110 → 111111110110
(two's complement)
12-bits
000000001001
+1
0000 0000 1010
= -10

slide 20

```
00000002: F1010104  ADD R1,#4
```

Cortex-M3 instructions are half-word aligned;
LDR & STR can access byte addr

slide 14

slide 9,15

Lecture 5/Homework