# Source Of The Week

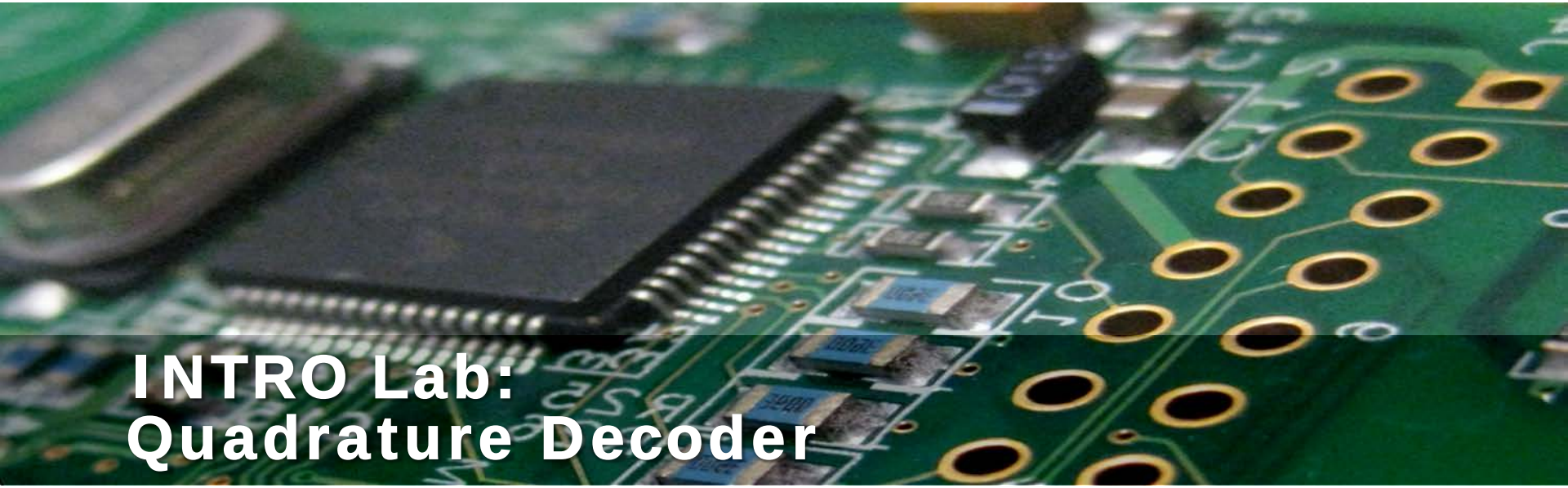
```
if (var1 == true)
{
    return true;
}
 else if (var1 == false)
{
    return false;
}
else
{
    return !true && !false;
}
```

# Source of the Day

```
void APP_Run(void)
 {
   static FIL fp;
   UINT a;
   char buffer[10] = "Help me!!";

   SHELL_Init();
   vTaskStartScheduler();
   (void)FAT1_open(&fp, "./test.txt",
                   FA_CREATE_ALWAYS|FA_WRITE);
   (void)FAT1_write(&fp, buffer, 10, &a);
   (void)FAT1_close(&fp);
 }
```

Lucerne University of
Applied Sciences and Arts
HOCHSCHULE
LUZERN
Technik & Architektur



# INTRO Lab:
# Quadrature Decoder

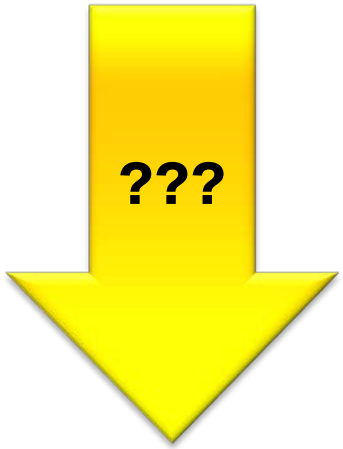
*“Dice and slice…”*

## Scriptum:
## Position Encoder

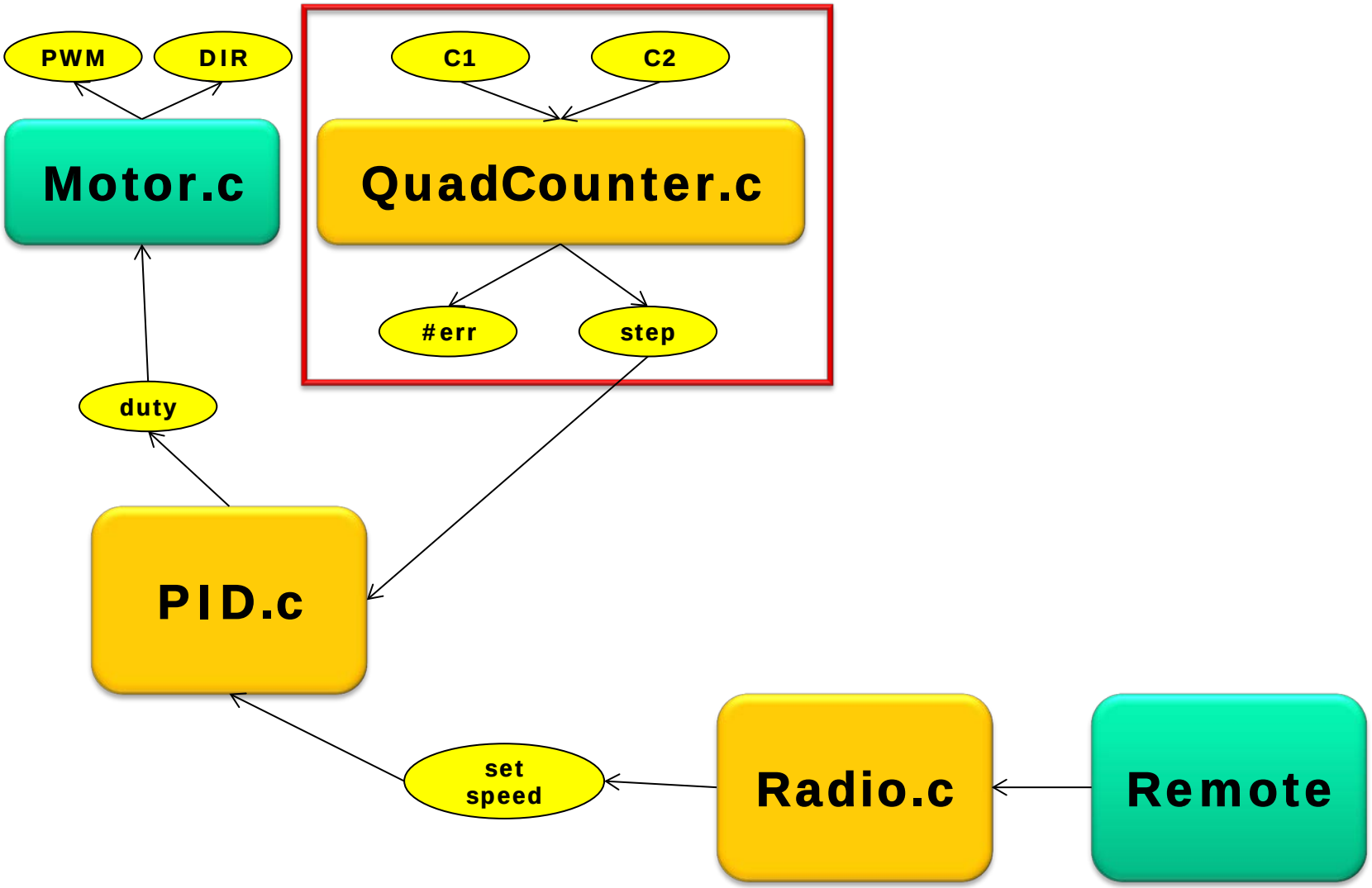
**Prof. Erich Styger**
*erich.styger@hslu.ch*
*+41 41 349 33 01*

# Learning Goals

- Quadrature decoding
- Real-time aspects
- Digital Signal Sampling
- Decoding
- Calculation Speed

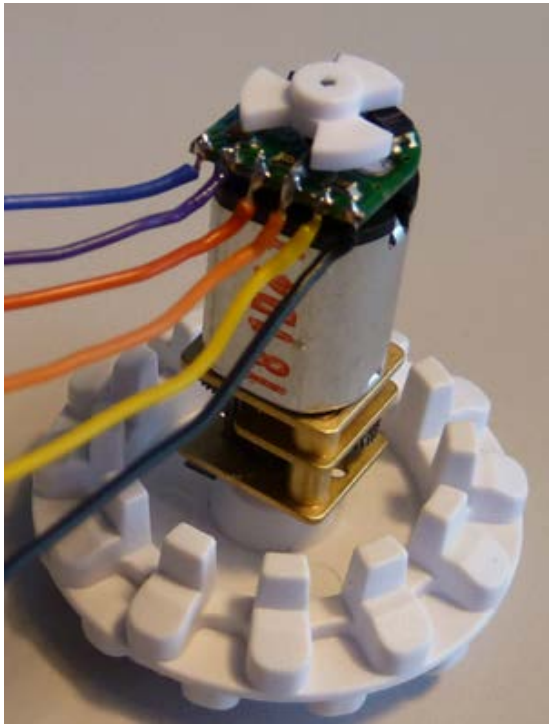


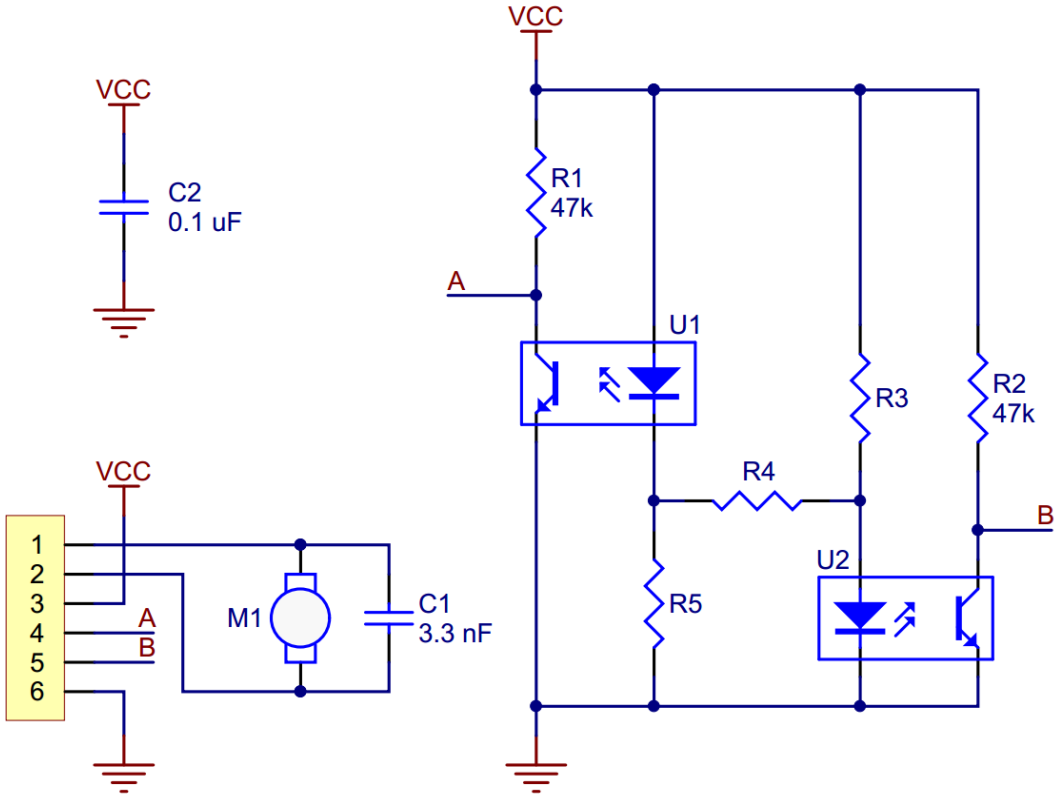
# High Level Overview



# Robo V1: Optical Motor Shaft Encoder

- 3 tooth: 3*4 steps per **shaft** revolution





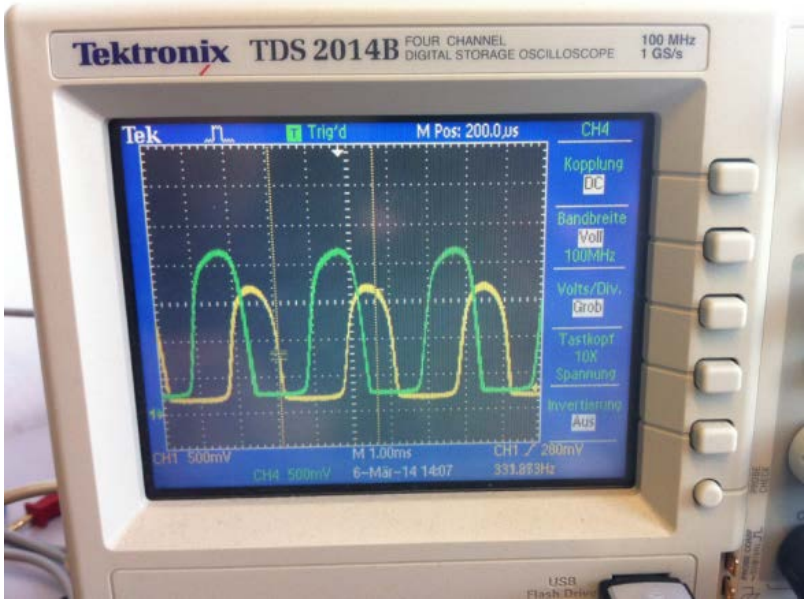

5 V version: R4 = 220 Ω, R3 and R5 not populated (emitters in series)

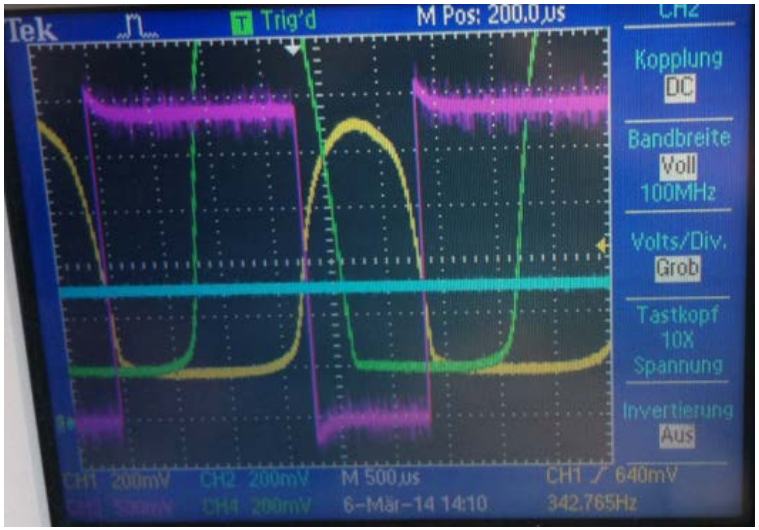
3.3 V version: R4 not populated, R3 and R5 = 180 Ω (emitters in parallel)

*Source: Pololu*

# Robo V1 Encoder Signals

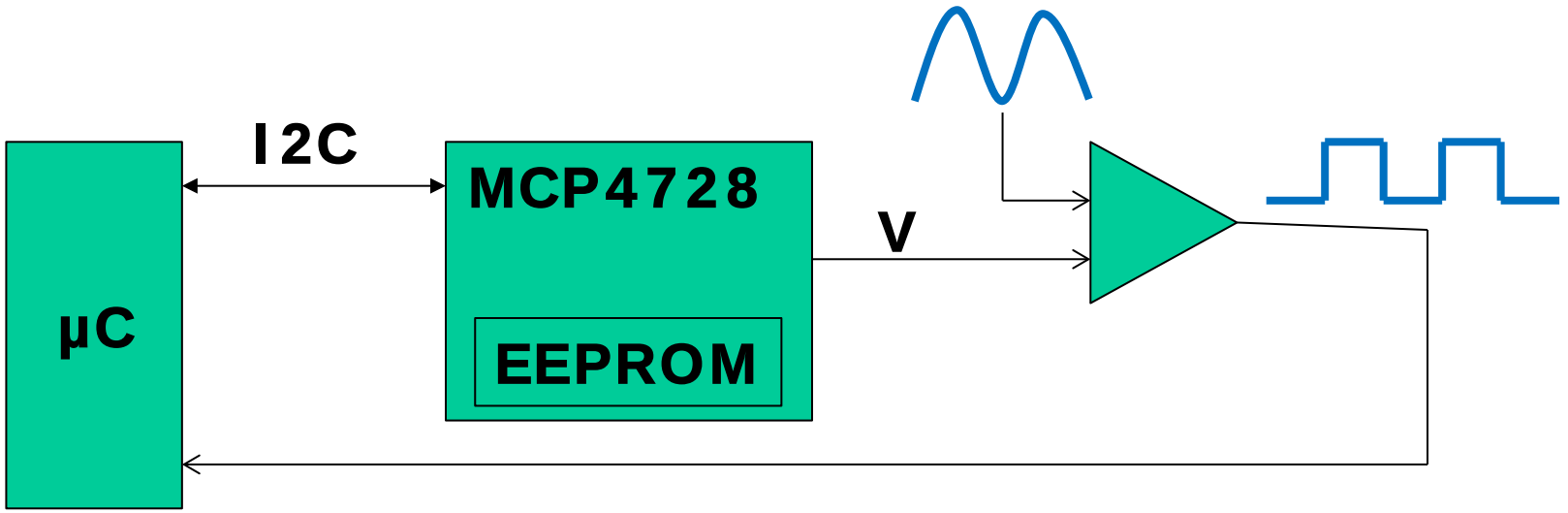
- Analog/Sinus-like signals
- Need to tune/calibrate
  - Signal Duty ration
  - Goal: 50%



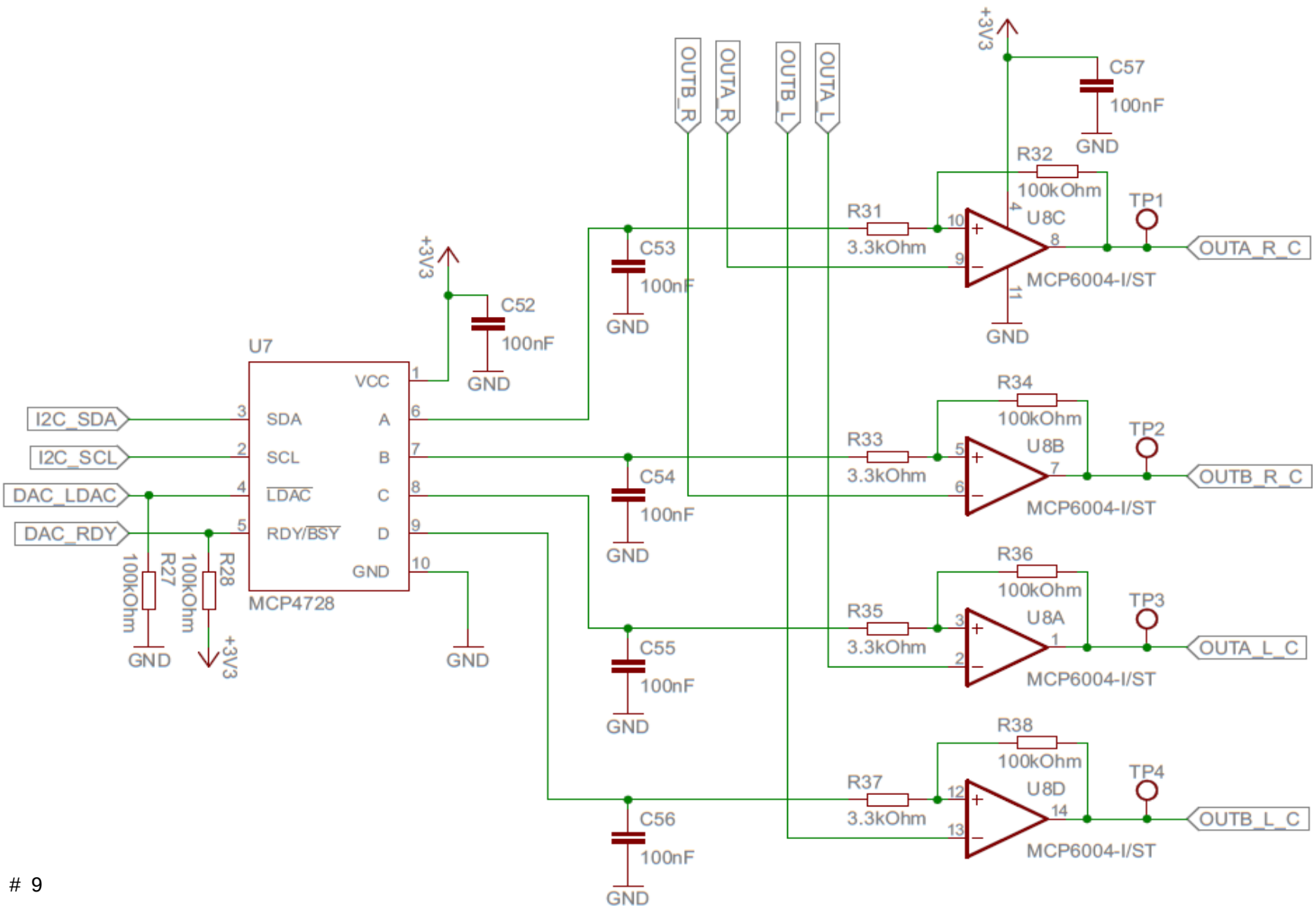



# Robo V1 Encoder Signal Processing

- D/A converter with comparator
- Infos
  - http://mcuoneclipse.com/2014/03/08/processing-the-pololu-motor-shaft-encoders/
- Lab robots: should be calibrated/tuned

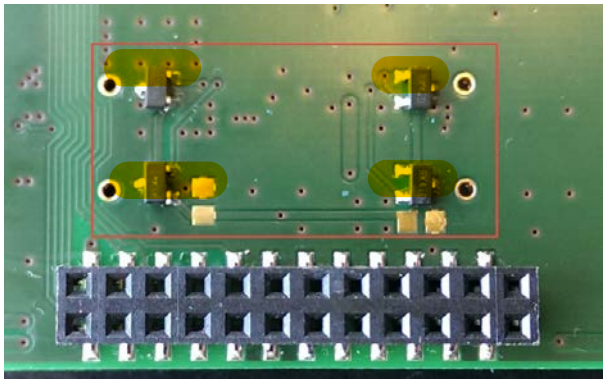


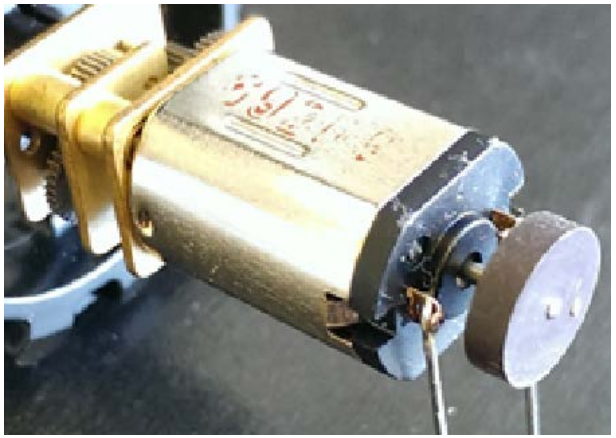
# Robo V1 Schematic

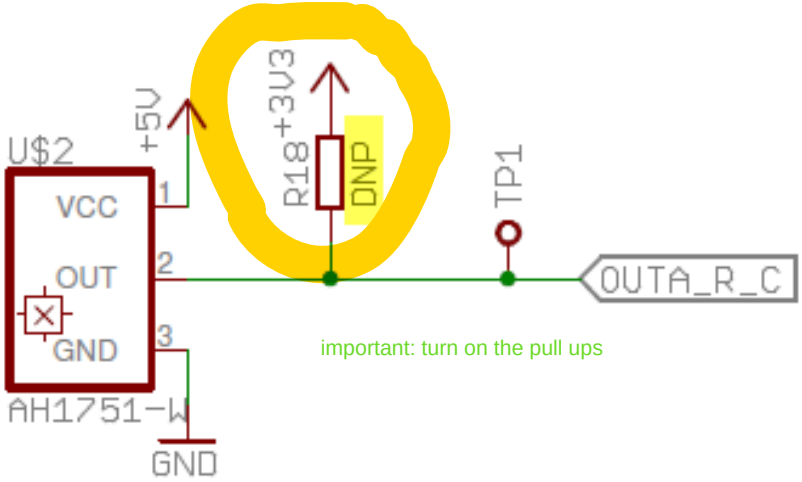


# Robo V2: Magnetic Encoders

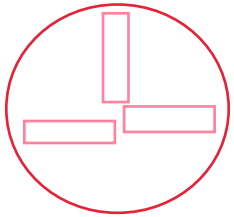
- Magnetic encoder with hall sensors
- Producing proper quadrature signals
- 3x4 ticks per revolution
- Need internal pull-ups







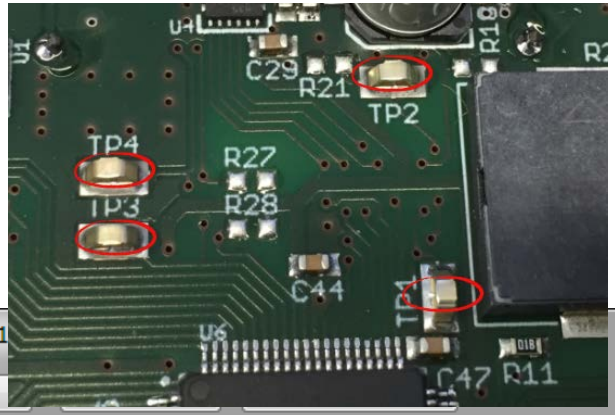

important: turn on the pull ups

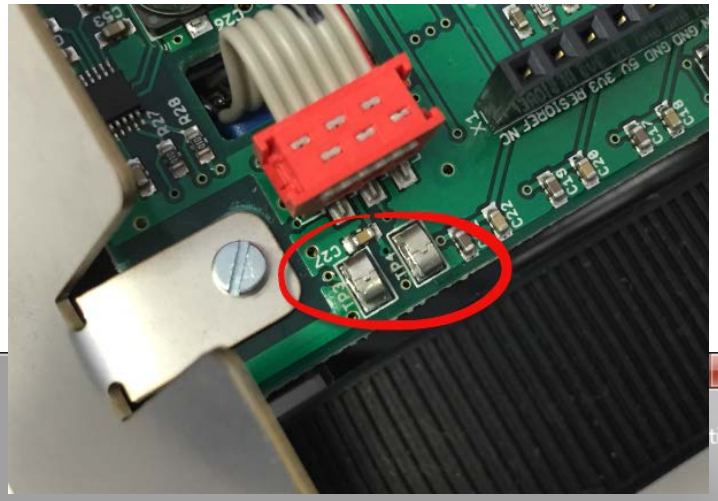


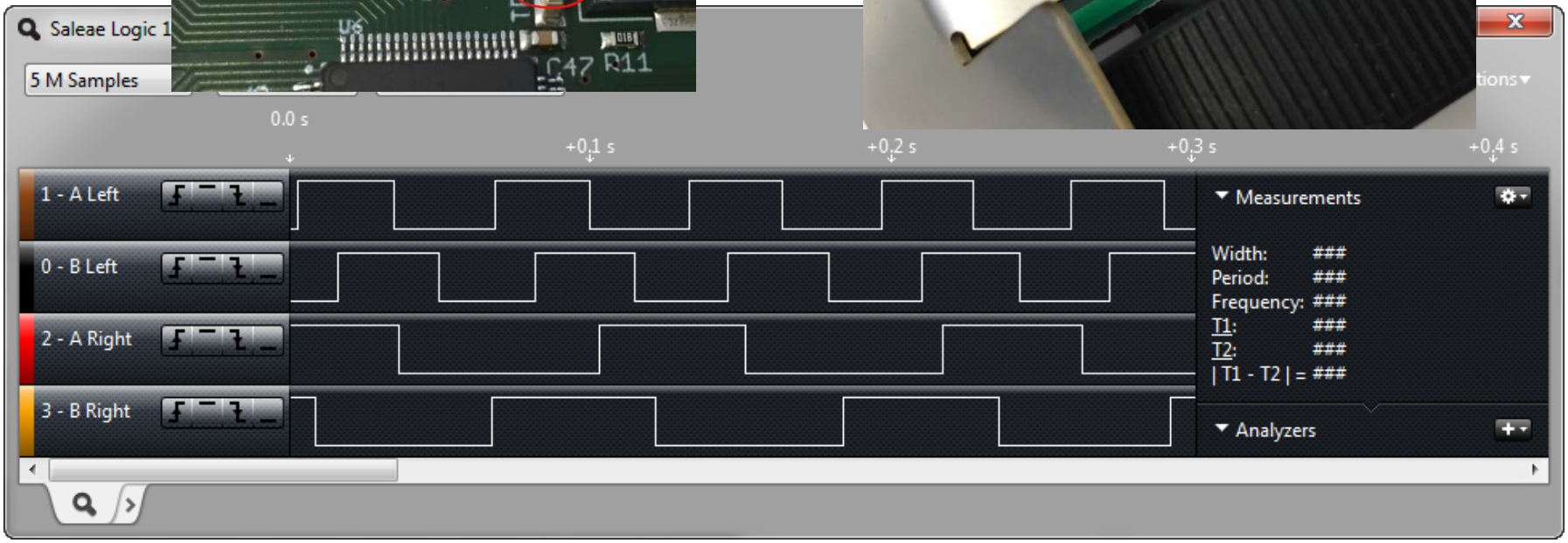
12 Zustände
12* 1:75 Auflösung

# Goal: 50% - 50% Signals

- Signals shifted by 90° (1 Quadrant)
- 50% Duty Cycle

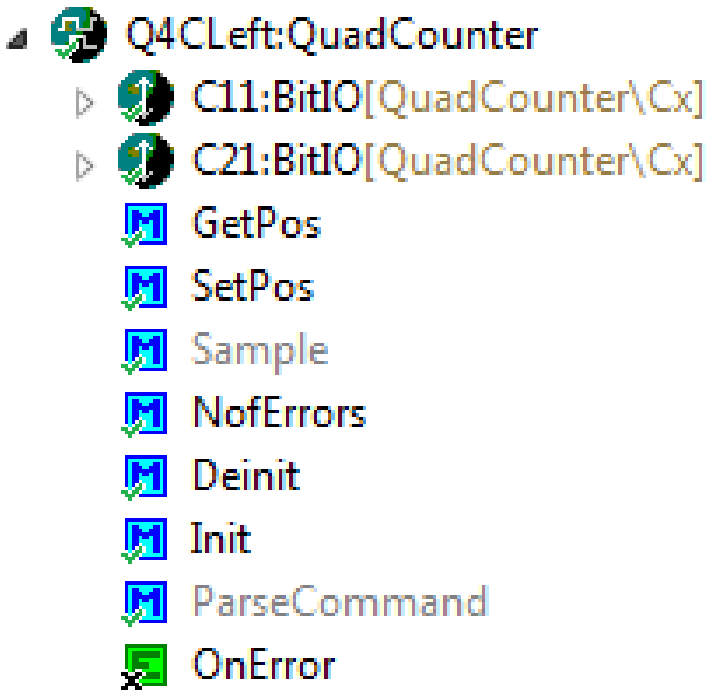


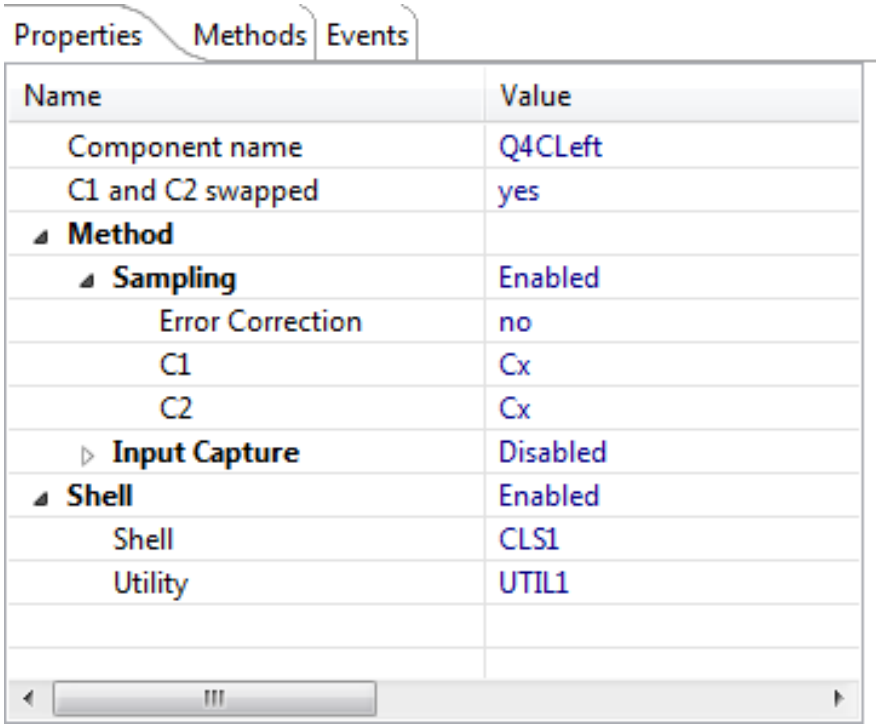





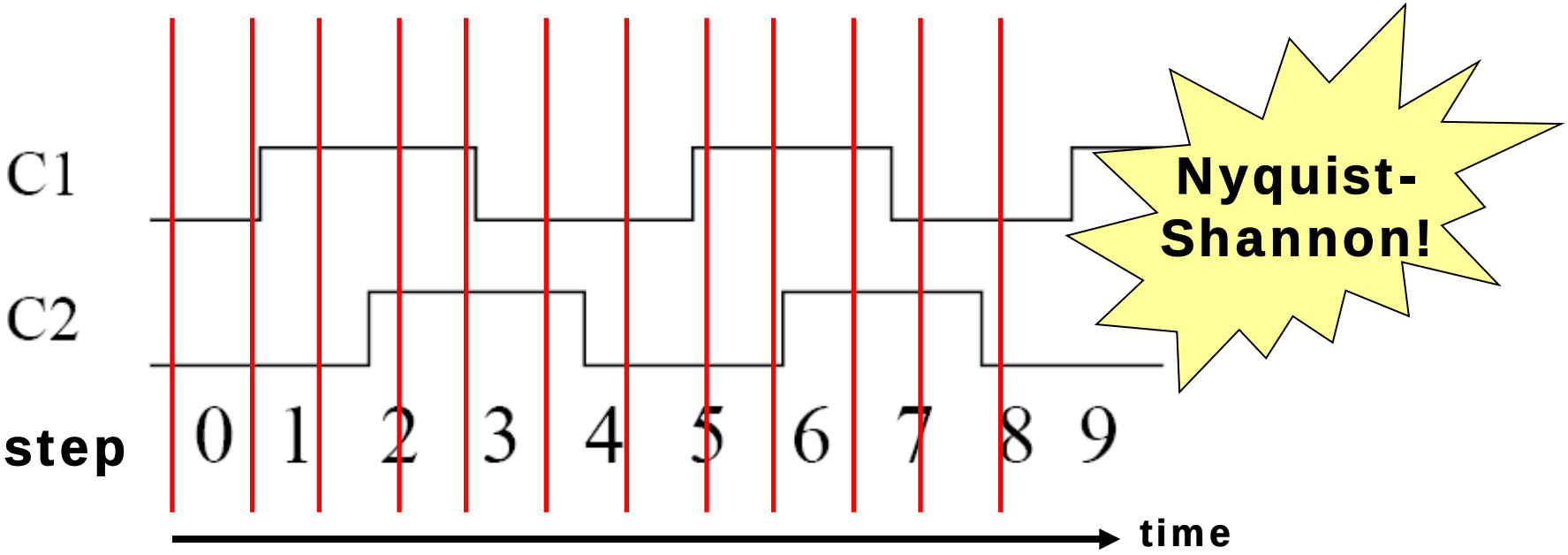
# QuadCounter Component

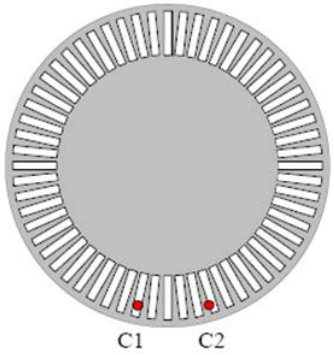
- **QuadCounter** Component
- Sampling: needs to be called periodically



# Quadrature Signal Sampling

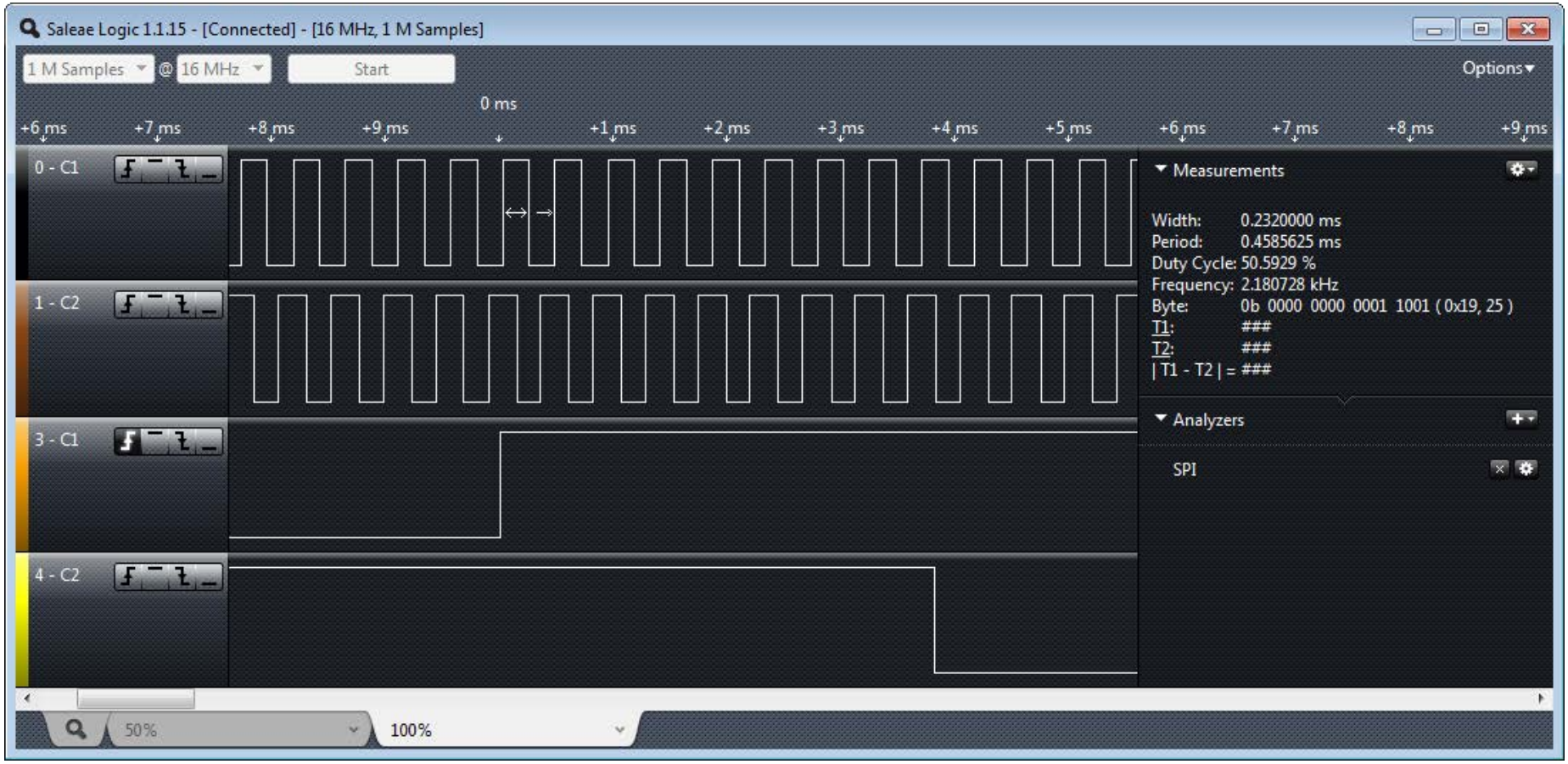


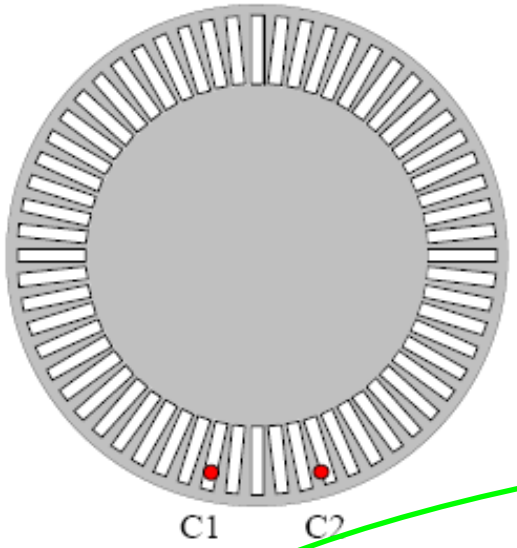

**100 holes/rev**
**4*100 Signals/s**
**1 rev/s → 1s/2*400 → 1.25 ms**

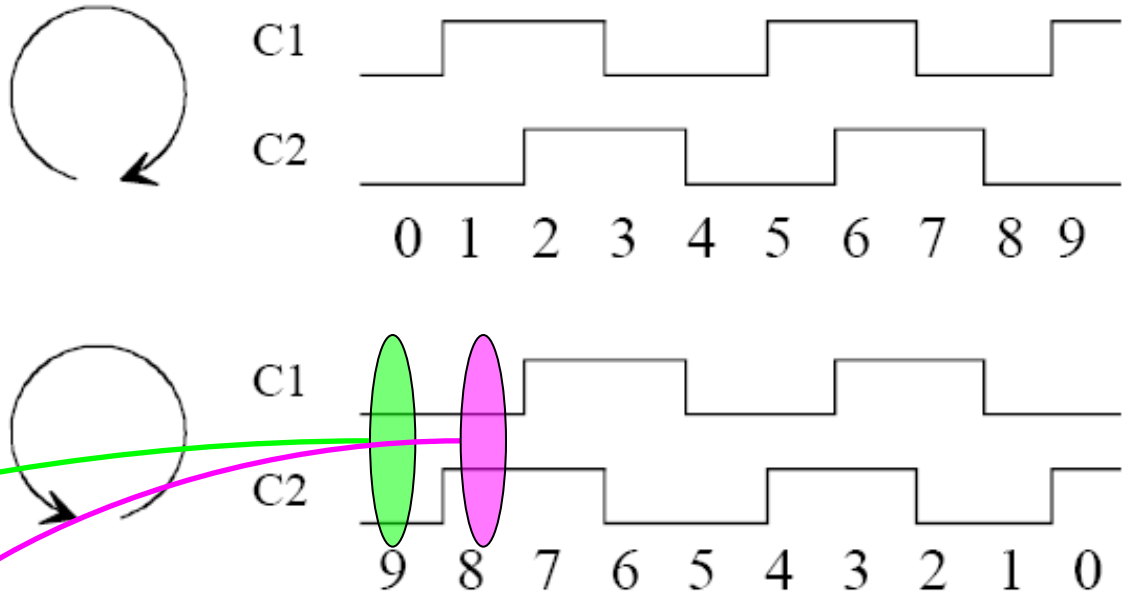
# Wheel Shaft Encoder

- Width: 232µs



# Quadrature Decoding

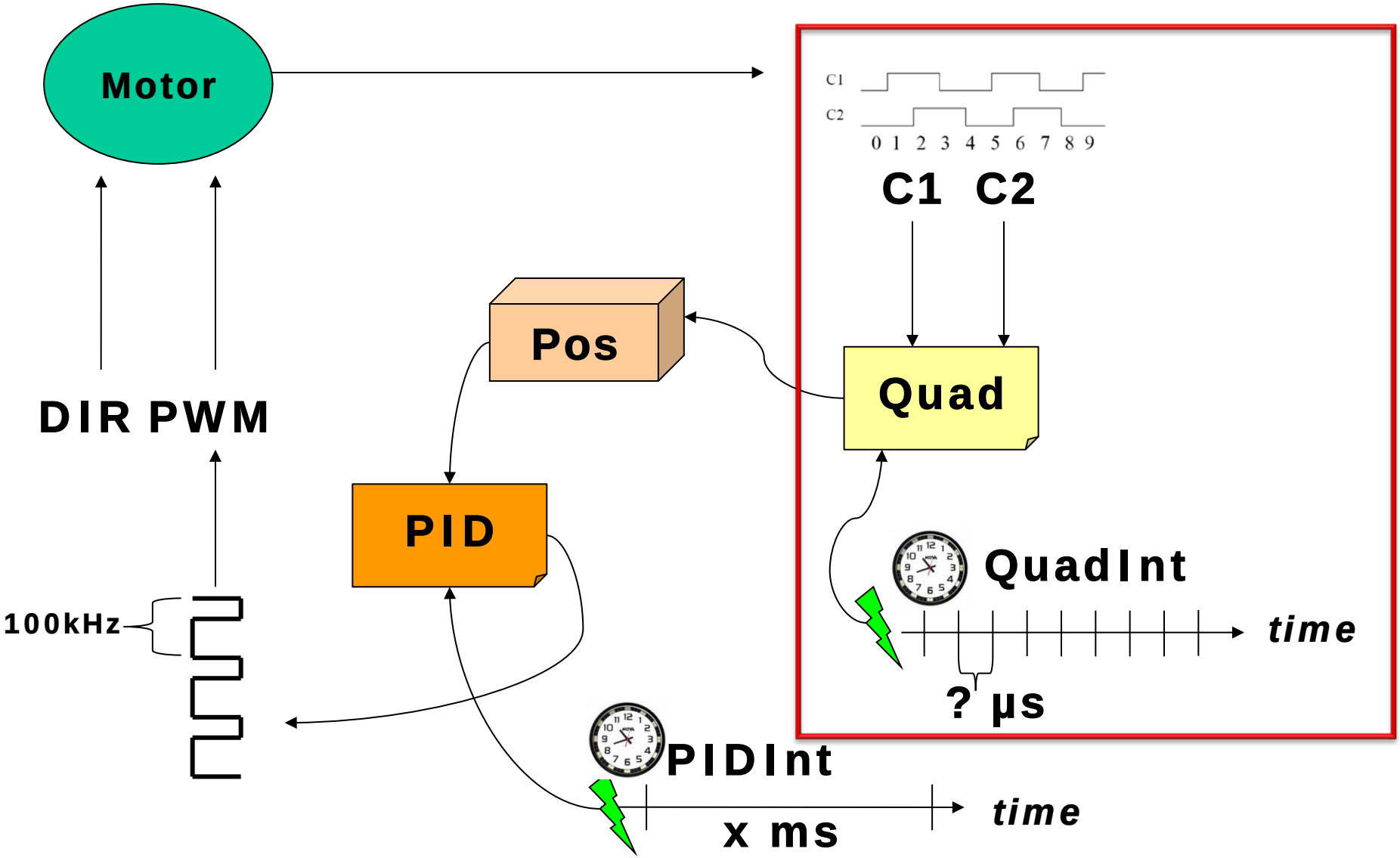


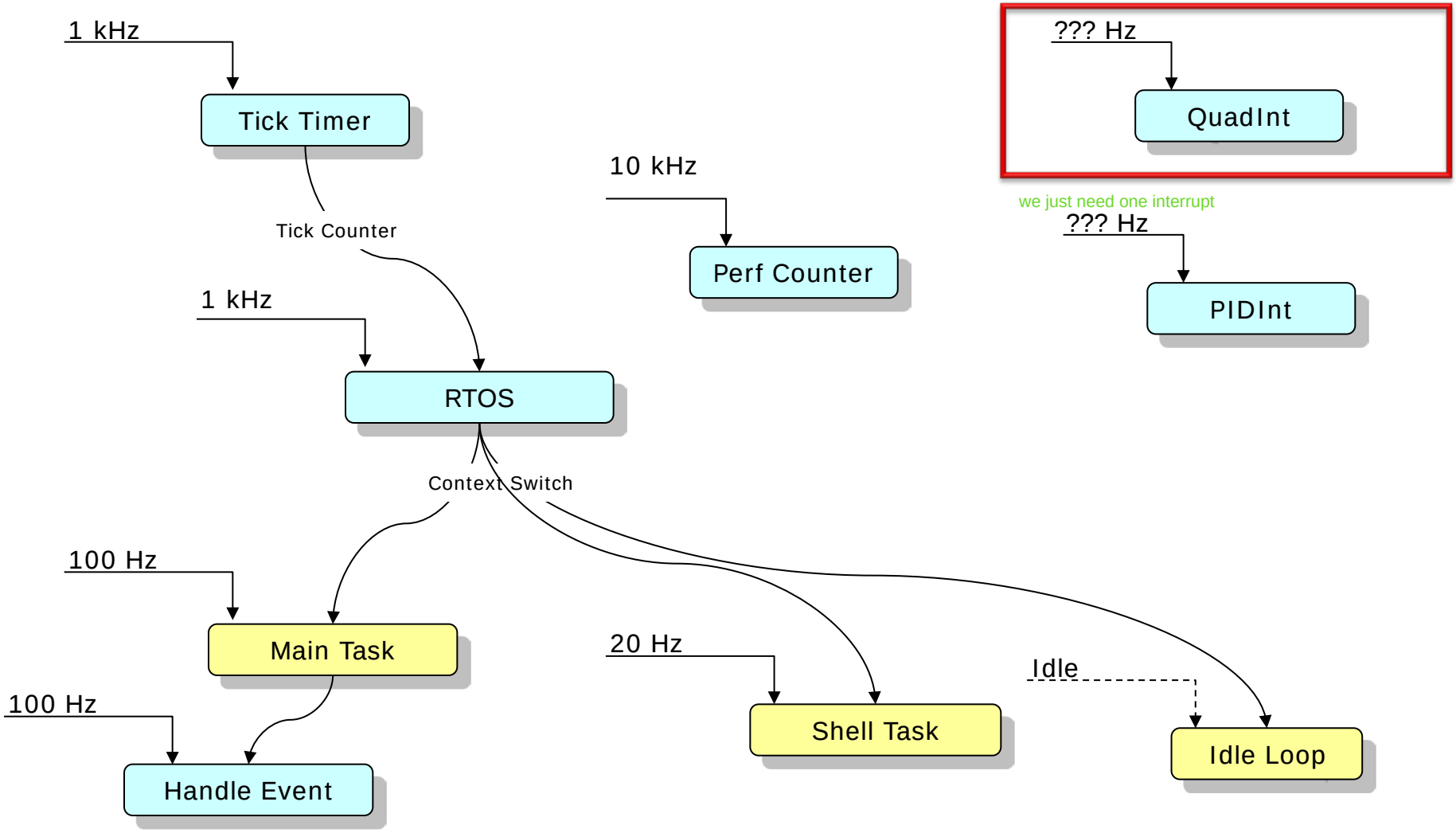

```
static const int8_t QUAD_Table[4][4] =
  {                     /* prev   new     */
    {                   /* c1 c2  c1 c2   */
     0,           [0]   /* 0  0   0  0   no change or missed a step? */
     1,           [1]   /* 0  0   0  1    */
     -1,          [2]   /* 0  0   1  0    */
     QUAD_ERROR   [3]   /* 0  0   1  1   error, lost impulse */
    },
```

[0]

# System Architecture & Timing



# Process Diagramm



# Real-time Aspects

- System load it's really a problem
  - Timer/RTOS
  - Quadrature Sampling

- Interrupts
  - As fast as possible
  - As short as possible
  - Avoid complex calculations/function calls

- Usage of
  - Oscilloscope
  - Logic Analyzer

# Lab: Quadrature

- V1 only (should already be calibrated):
  - MCP4728
- Add QuadCounter components
  - Review/Understand
- Quadrature Decoding
  - QuadInt Timer Interrupt
  - Determine QuadInt timer period
  - Steps + Direction ➔ step counter
  - Error rate
- Shell Integration
  - Steps
  - Errors