Quadrature Decoding

ENCE361 Embedded Systems 1

Course Coordinator: Ciaran Moore (ciaran.moore@Canterbury.ac.nz)

Lecturer: Le Yang (le.yang@canterbury.ac.nz)

Department of Electrical and Computer Engineering

Where we’re going today

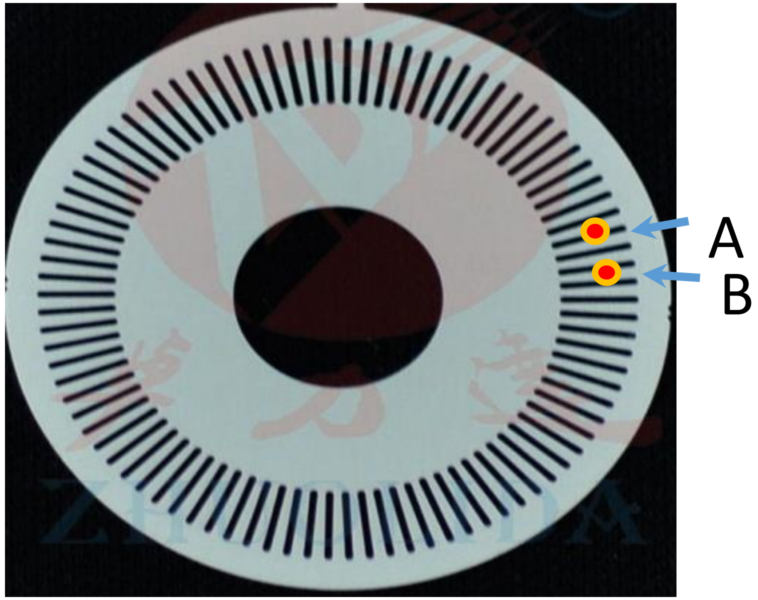
- **Quadrature decoding overview**
- Software quadrature decoding
- Absolute position determination
- Homework

Quadrature Decoding Overview (1)

- Quadrature in math
  - The process of constructing a square having the same area of a figure

- Quadrature in electrical engineering
  - Two sinusoids that are offset in phase by $\frac{\pi}{2}$ (90^o)
  - $\sin(2\pi f t)$ and $\sin\left(2\pi f t + \frac{\pi}{2}\right) = \cos(2\pi f t)$

- Determine distance and direction of rotation
  - With slotted disk and a pair of optical detectors

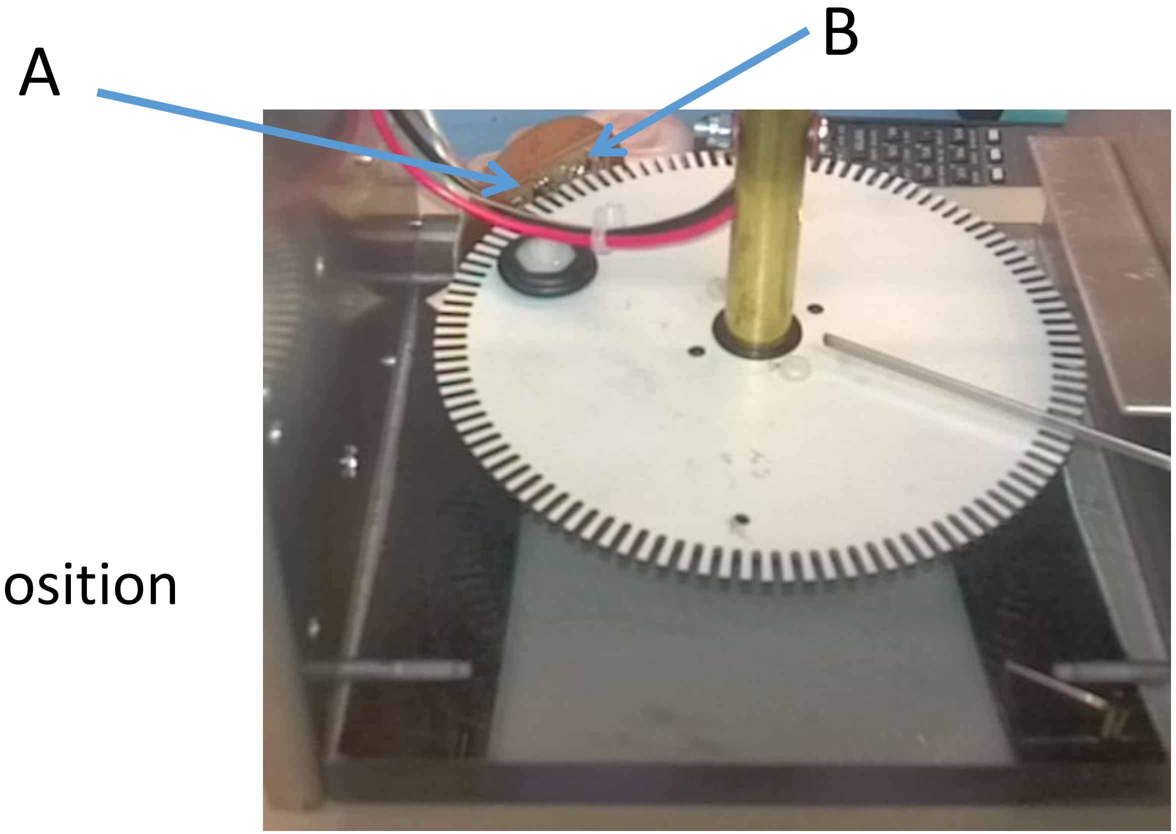


Quadrature Decoding Overview (2)

- Helicopter yaw transducer
  - Channel A: J1-03 (PB0)
  - Channel B: J1-04 (PB1)

  - Sensors A and B:
    - Rotation direction and relative position
    - Quadrature decoding needed

  - Need a starting datum for absolute position



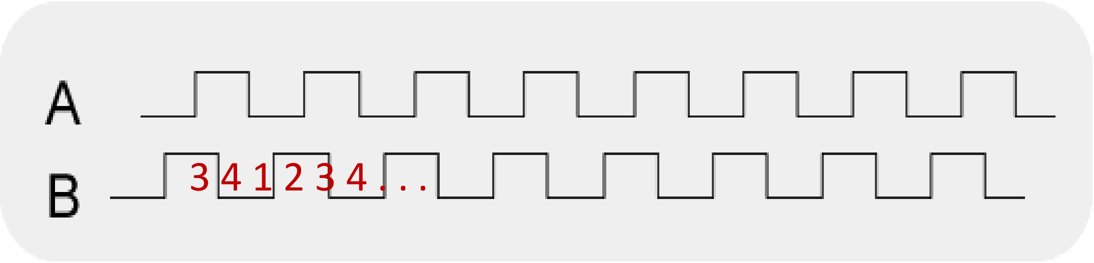
Quadrature Decoding Overview (3)

- Slotted disk rotation makes signals at sensors A and B be 90^o out of phase

Clockwise rotation

B leads A

Phase	A	B
1	0	0
2	0	1
3	1	1
4	1	0



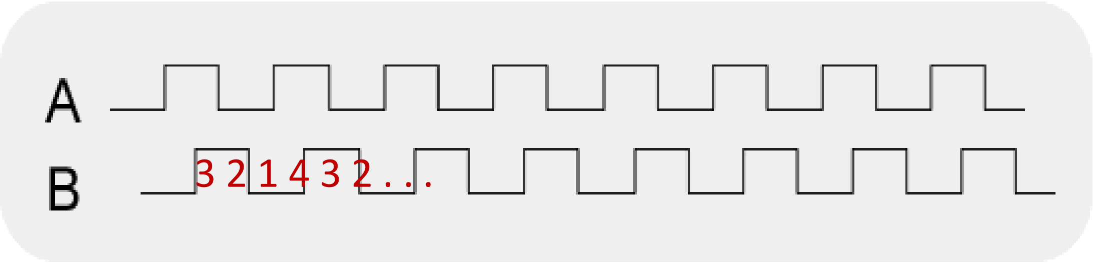
B leads A

Time

Anticlockwise rotation

A leads B

Phase	A	B
4	1	0
3	1	1
2	0	1
1	0	0



A leads B

Where we’re going today

- Quadrature decoding overview
- **Software quadrature decoding**
- Absolute position determination
- Homework

Software Quadrature Decoding (1)

- Software quadrature decoding using finite state machine (FSM)

B leads A

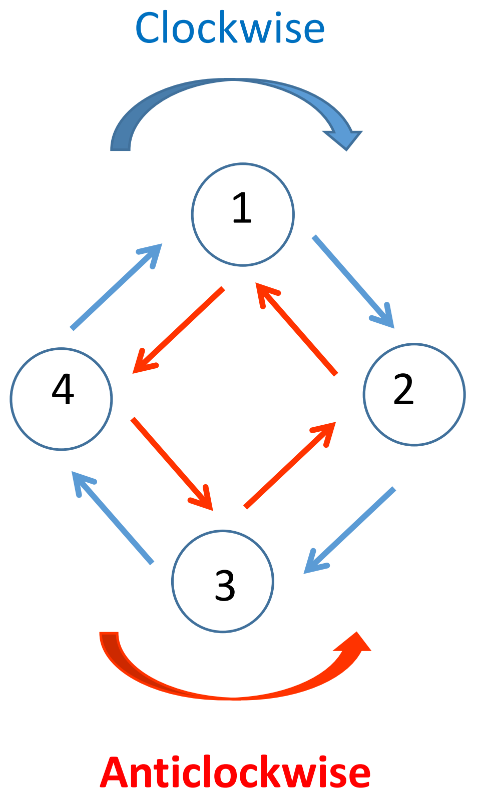
Clockwise rotation

Phase	A	B
1	0	0
2	0	1
3	1	1
4	1	0

A leads B

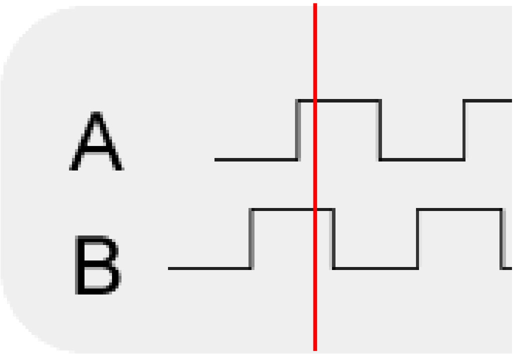
Anticlockwise rotation

Phase	A	B
4	1	0
3	1	1
2	0	1
1	0	0

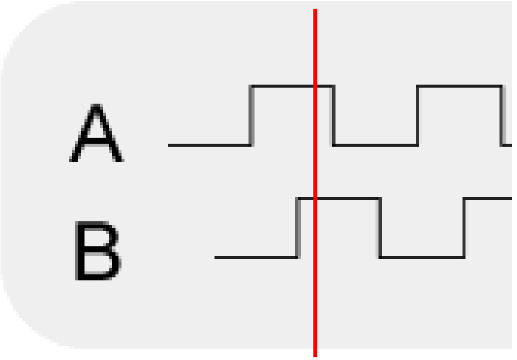


Software Quadrature Decoding (2)

- Pseudocode for a possible implementation of FSM on Slide 7



Clockwise

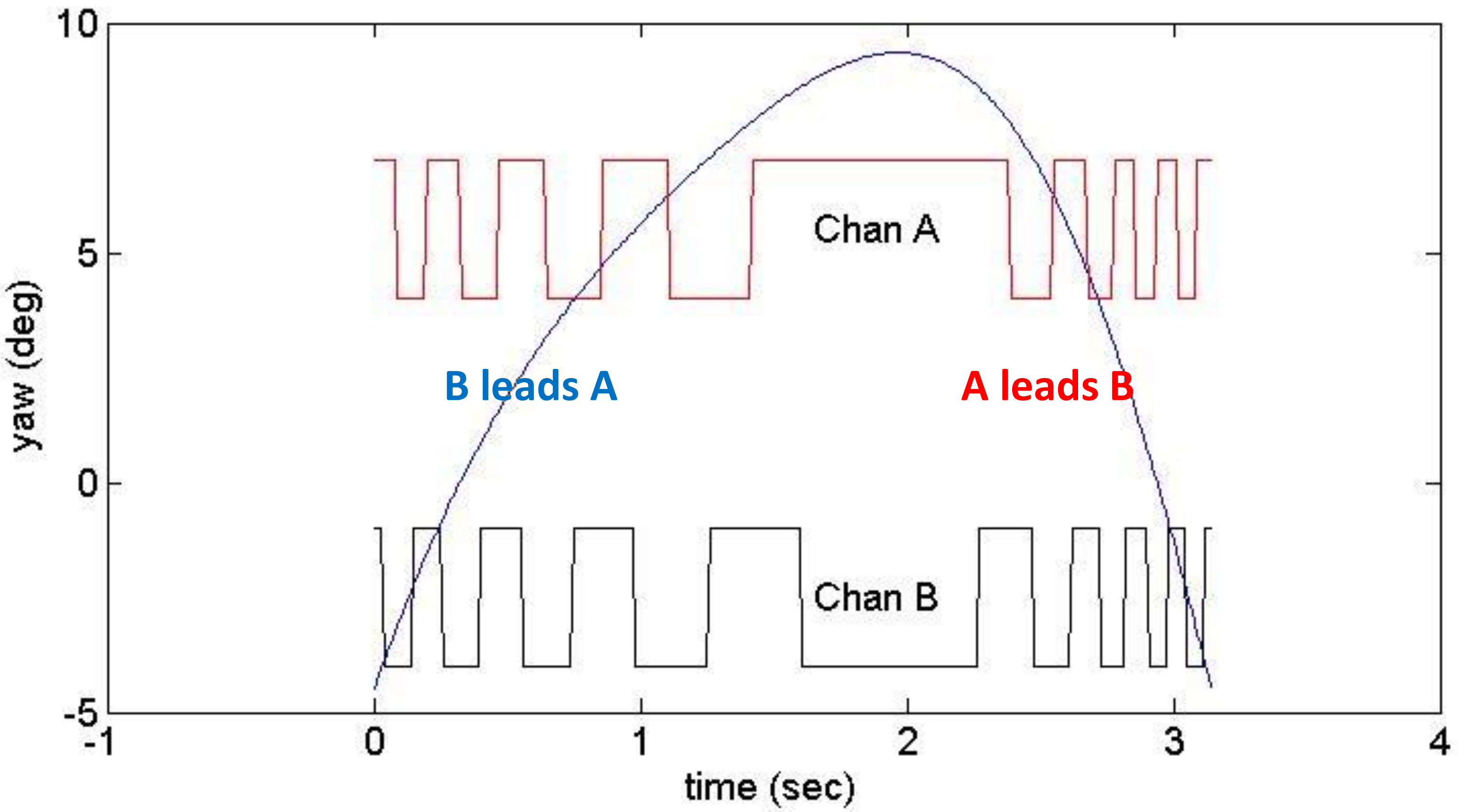


Anticlockwise

```
state (AB) = 11;              // Current State
if (next change is on channel B)
{
    state (AB) = 10;
    yaw++;                    // clockwise
}
else                          // next change is on channel A
{
    state (AB) = 01;
    yaw--;                    // counterclockwise
}
```

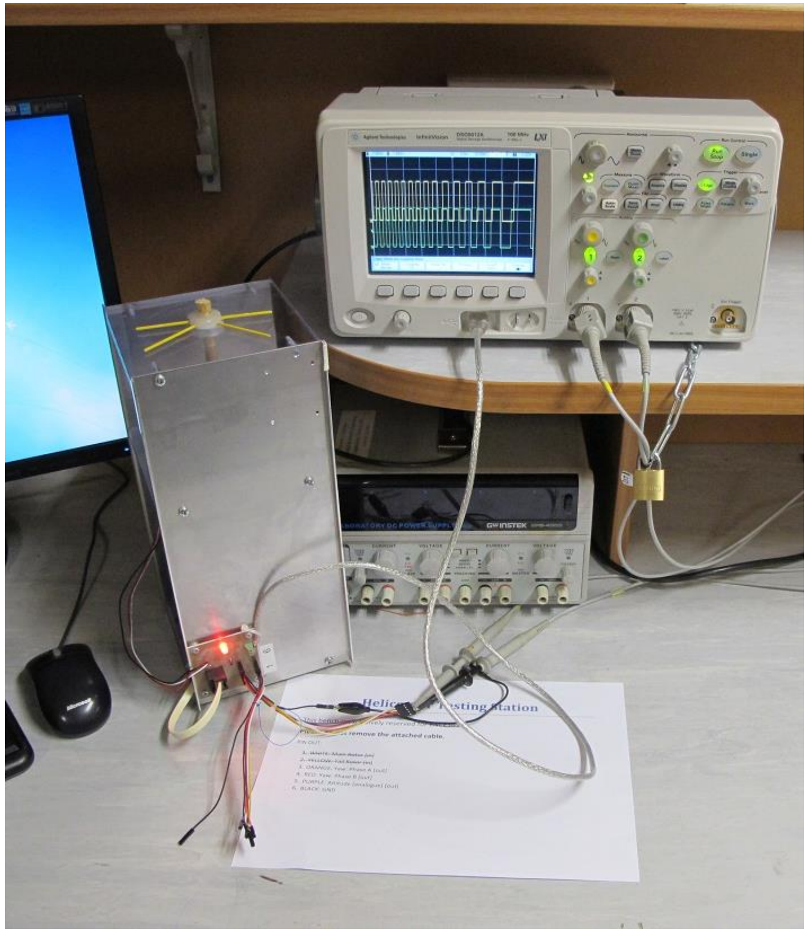
Example Yaw Trajectory

- Helicopter rotates clockwise (yaw increases)
- Helicopter rotates anticlockwise (yaw decreases)



Testing Setup

- Debug your program with a helicopter mount unit and oscilloscope



Quadrature Encoder on Tiva MCU (1)

- Tiva MCU has two quadrature encoder with index (QEI) modules
  - Each module interprets the code from a quadrature encoder wheel
    - Programmable noise filter on the input with a frequency as high as ¼ of MCU frequency
  - Determine rotation direction
    - Quadrature decoding
  - Integrate position over time → Distance
  - Find running estimate of the wheel rotation speed
    - Use build-in timer
  - Interrupt generation
    - Velocity-timer expiration
    - Direction change
    - Quadrature error detection

Quadrature Encoder on Tiva MCU (2)

- Some API function prototypes in driverlib/qei.h

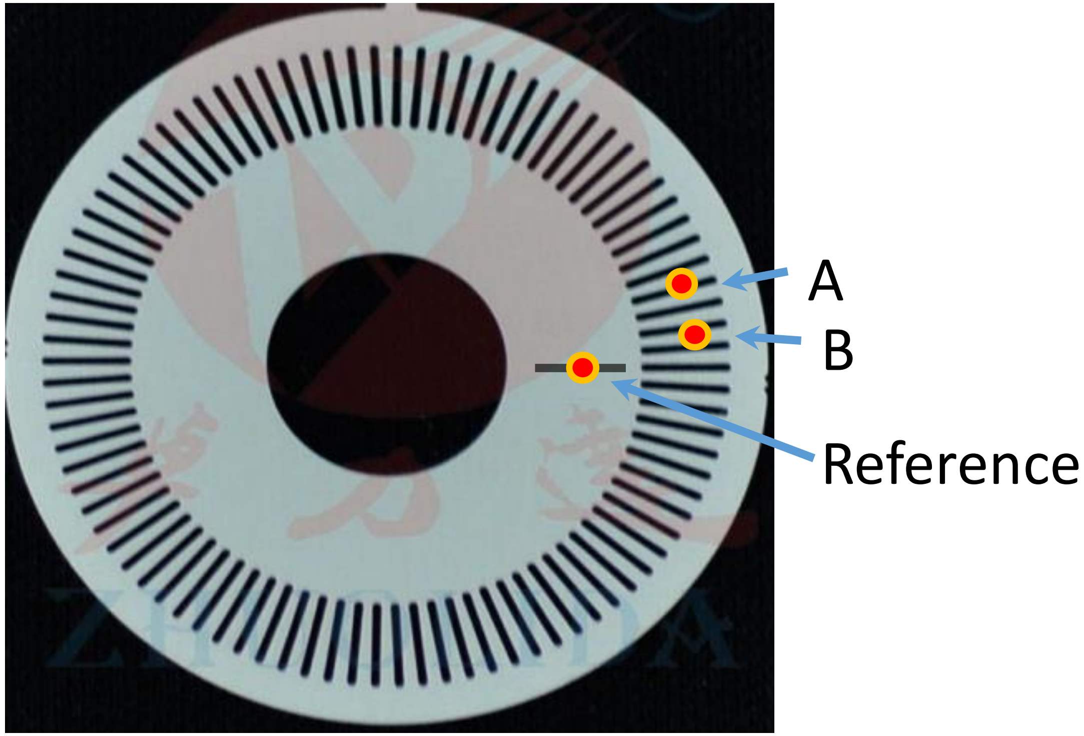
```
void QEIConfigure (unsigned long ulBase, unsigned long ulConfig, unsigned long ulMaxPosition)
long QEIDirectionGet (unsigned long ulBase)

void QEIIntClear (unsigned long ulBase, unsigned long ulIntFlags)
void QEIIntDisable (unsigned long ulBase, unsigned long ulIntFlags)
void QEIIntEnable (unsigned long ulBase, unsigned long ulIntFlags)
void QEIIntRegister (unsigned long ulBase, void (*pfnHandler)(void))

unsigned long QEIPositionGet (unsigned long ulBase)

void QEIVelocityConfigure (unsigned long ulBase, unsigned long ulPreDiv, unsigned long ulPeriod)
void QEIVelocityDisable (unsigned long ulBase)
void QEIVelocityEnable (unsigned long ulBase)
unsigned long QEIVelocityGet (unsigned long ulBase)
```

Where we’re going today

- Quadrature decoding overview
- Software quadrature decoding
- **Absolute position determination**
- Homework

Absolute Position from Reference Sensor

- Rotate the helicopter by driving the tail motor to the reference position
    - HIGH input from the yaw reference signal



Homework

1. Label the transitions between states on the state transition diagram on Slide 7.

2. Why are there no transitions shown between opposing states, i.e. between 1 and 3, or between 2 and 4 on Slide 7?

3. If a quadrature encoded disk/sensor unit of the type shown on Slide 3 has 100 slots and it rotates at up to 10 revolutions per second, what is the minimum time between two interrupts generated by changes on the signals? Have you made any assumptions?

4. If a quadrature encoded disk/sensor unit of the type shown on Slide 3 has 100 slots and it rotates at up to 10 revolutions per second, at what minimum rate would the GPIO port for the two channel inputs have to be polled to guarantee that the motion could be correctly determined?