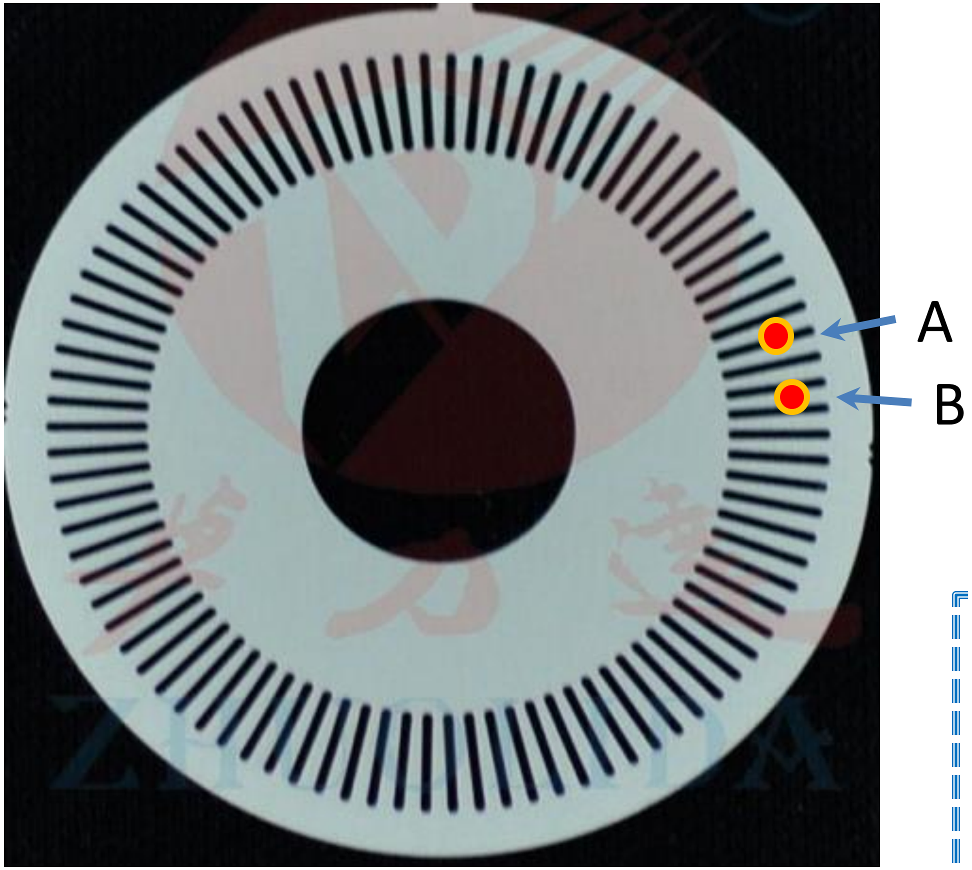
ENCE361 Embedded Systems 1   2018

Lecture 18

Quadrature decoding

Quadrature (de)encoding



Slotted disk and pair of optical detectors

quadrature

In maths: historical term which means determining area

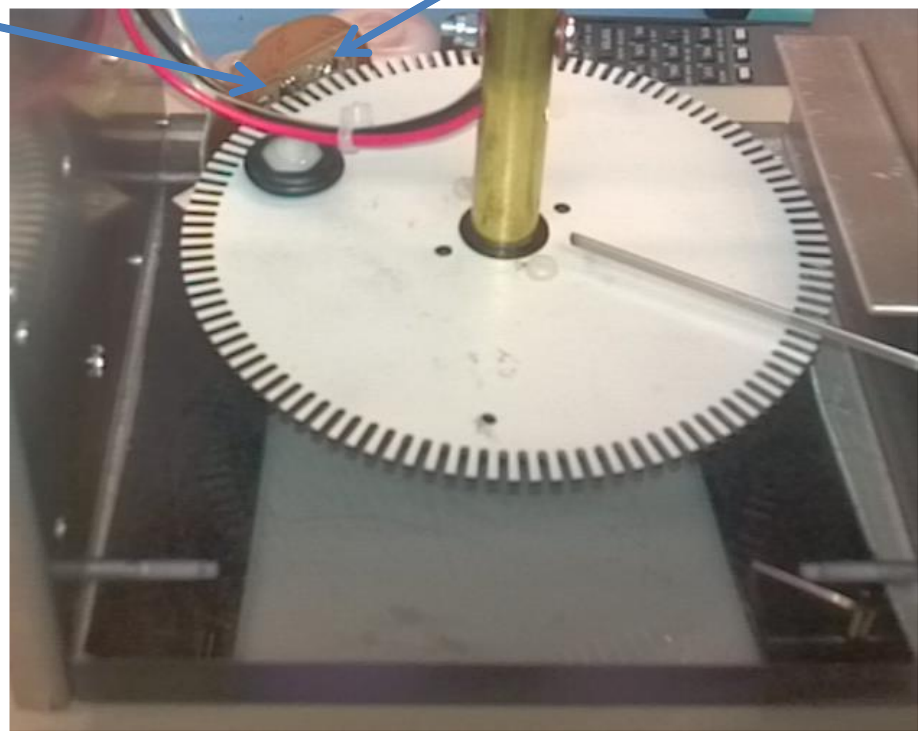
In EE: two sinusoids that are offset in phase by one-quarter cycle (π/2 radians).

Quadrature (de)encoding

Using a **relative position** transducer requires knowledge of **direction of rotation** and a **starting datum**

B

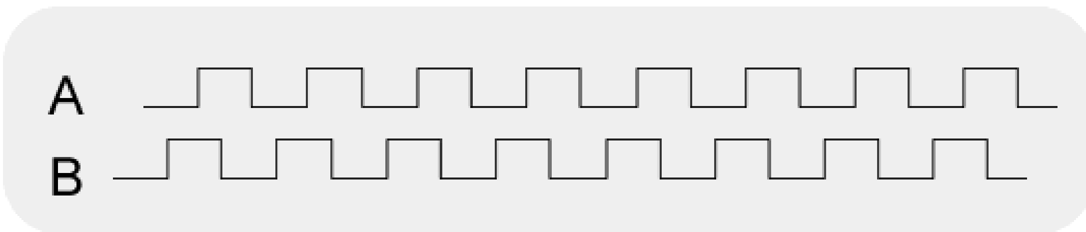
A



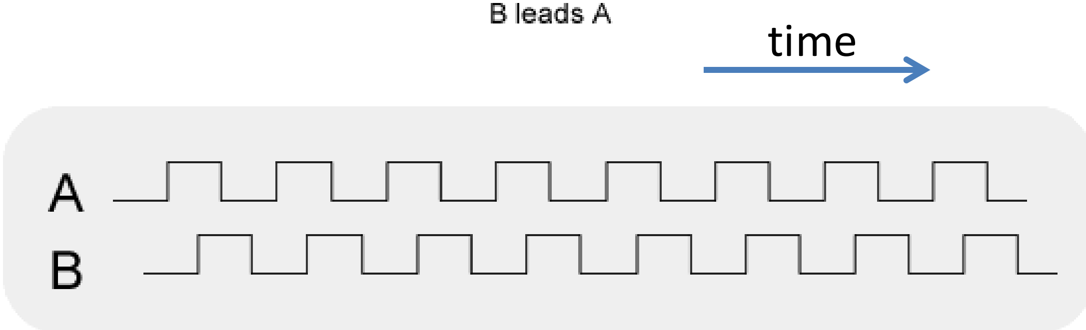
The helicopter yaw transducer used for ENCE361

Pin J1-03 (PB0) is ch A, Pin J1-04 (PB1) is ch B, i.e., Port B = 0000 00BA

Quadrature (de)encoding



B leads A



A leads B

clockwise rotation

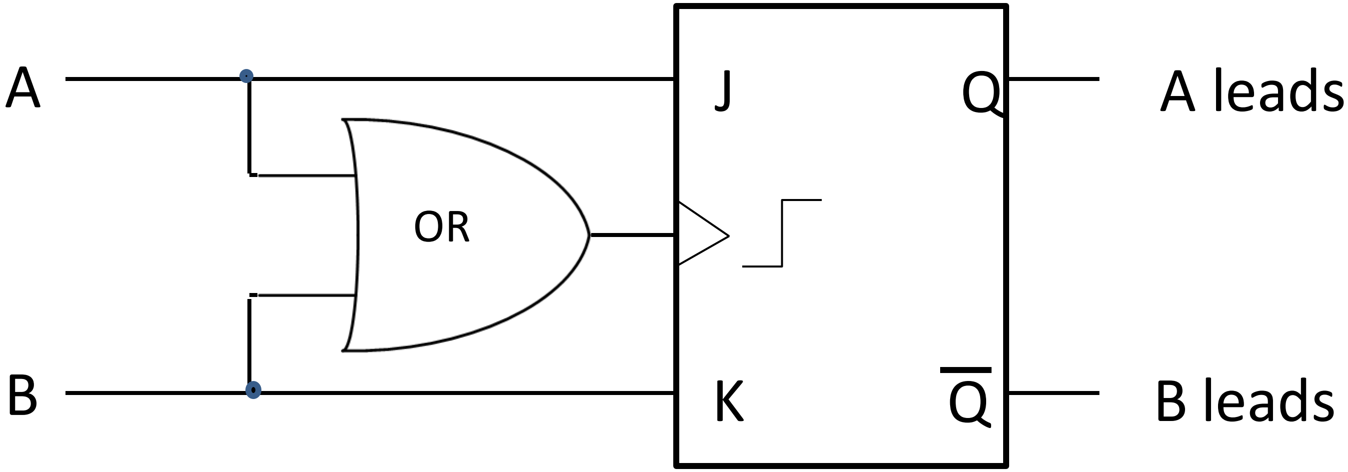
Phase	A	B
1	0	0
2	0	1
3	1	1
4	1	0

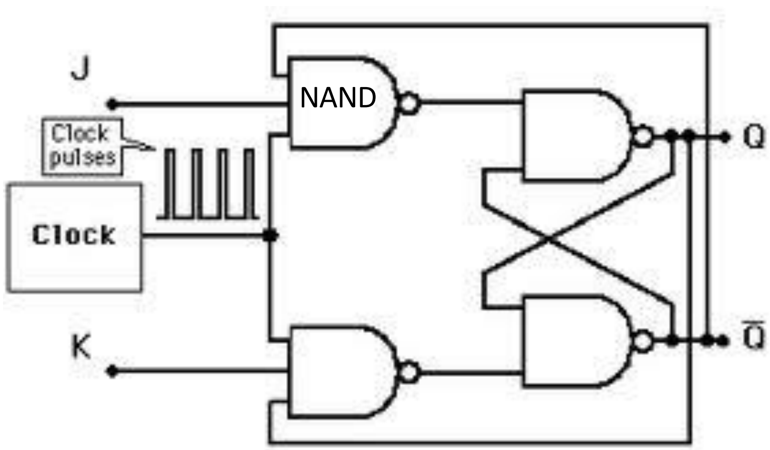
anticlockwise rotation

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

This depicts quadrature encoding with two sensors A and B positioned so that transitions due to motion of the stripes/slots occur 90 degrees out of phase.

Quadrature decoding with a J-K flip-flop





JK Flip Flop

Characteristic table

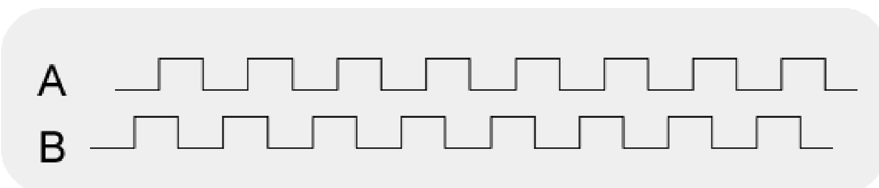
J	K	Q_{next}	Comment
0	0	Q	hold state
0	1	0	reset
1	0	1	set
1	1	$\overline{Q}$	toggle

Quadrature decoding as a state machine

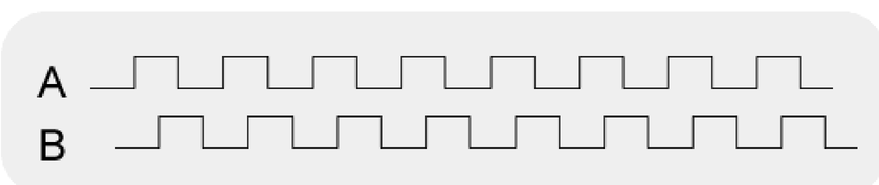
[Second example of a state machine.]

Monitor the channel A and B inputs from the quadrature angle encoder

- Pin-change interrupts sense when either channel A or B changes
- Can be represented as a 4-state finite state machine
- Arbitrary labels ‘1’, ‘2’, etc. have been assigned to the states



B leads A



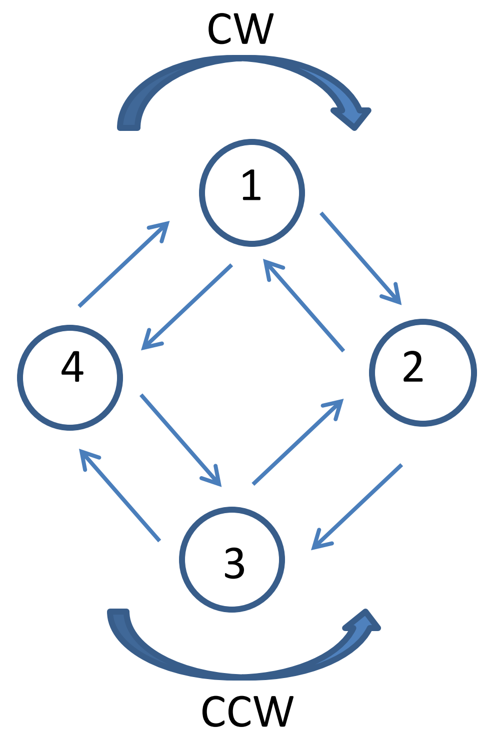
A leads B

clockwise rotation

State	A	B
1	0	0
2	0	1
3	1	1
4	1	0

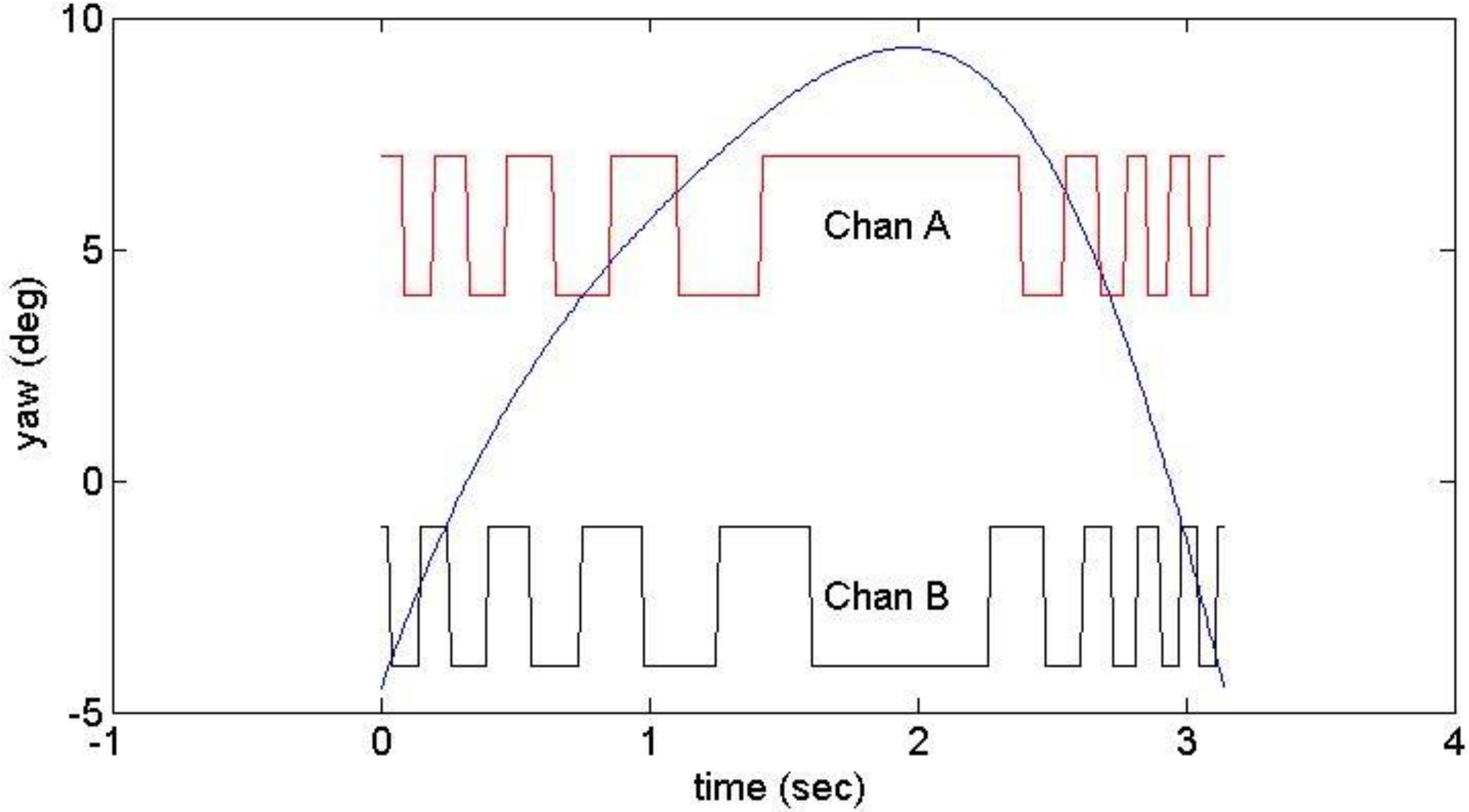
anticlockwise rotation

State	A	B
4	1	0
3	1	1
2	0	1
1	0	0



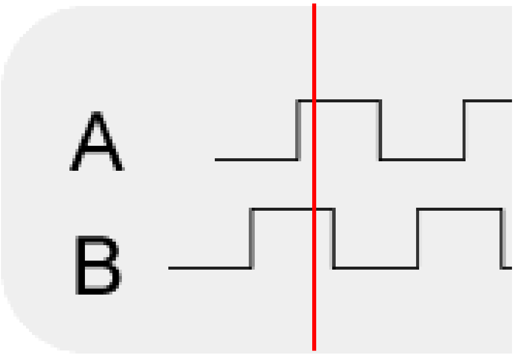
Example yaw trajectory

Helicopter rotates CW (yaw increasing), then CCW (yaw decreasing).

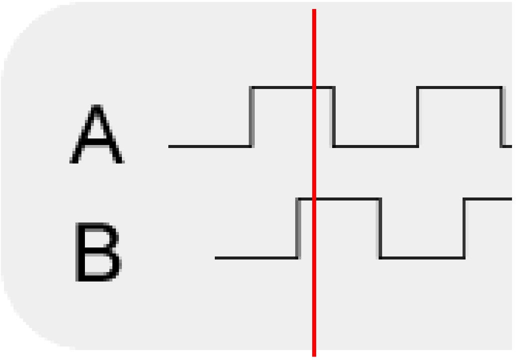


How can the state machine be implemented?

Quadrature decoding



CW



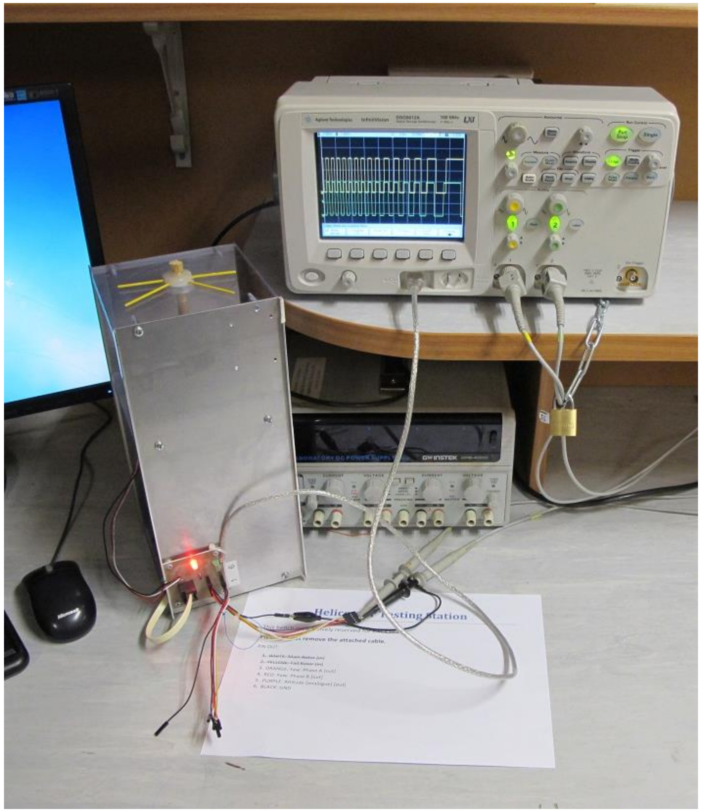
CCW

A fragment of pseudocode illustrating part of a possible implementation based on the FSM on Slide 6:

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

Testing setup for Milestone 2

Six helicopter mount units for manual use will be available (yaw and height): shared between ESL and Electronics Lab). **They must not be moved elsewhere**.



Instructions for connecting will accompany each unit

Quadrature Encoder units in the Tiva:

The Tiva TM4C123GH6PM microcontroller includes two "quadrature encoder with index" (QEI) modules. Each QEI module interprets the code produced by a quadrature encoder wheel to integrate position over time and determine direction of rotation. In addition, it can capture a running estimate of the velocity of the encoder wheel.

Each Tiva® quadrature encoder has the following features:

- ■ Position integrator that tracks the encoder position
- ■ Programmable noise filter on the inputs
- ■ Velocity capture using built-in timer
- ■ The input frequency of the QEI inputs may be as high as 1/4 of the processor frequency (for example, 12.5 MHz for a 50-MHz system)
- ■ Interrupt generation on:
  - – Index pulse
  - – Velocity-timer expiration
  - – Direction change
  - – Quadrature error detection

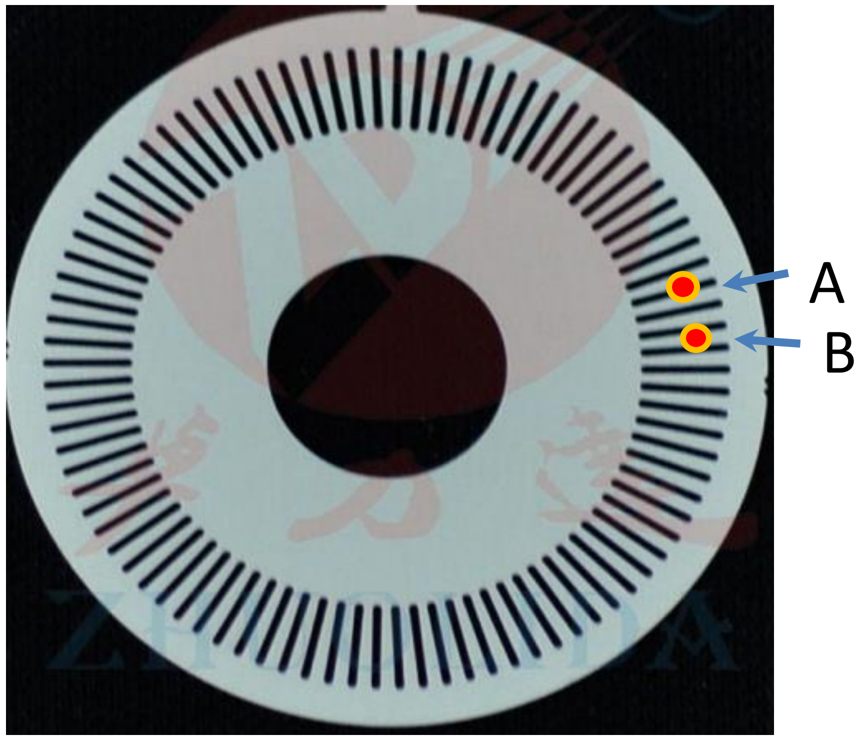
Remember that you are not using the Tiva QEI units in the Project

Quadrature Encoder units in the TM4C123GH6PM:

`driverlib/qei.h` has the following API prototypes:

```
void QEIConfigure (unsigned long ulBase, unsigned long ulConfig,
    unsigned long ulMaxPosition)
long QEIDirectionGet (unsigned long ulBase)
void QEIDisable (unsigned long ulBase)
void QEIEnable (unsigned long ulBase)
tBoolean QEIErrorGet (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 QEIIntStatus (unsigned long ulBase, tBoolean bMasked)
void QEIIntUnregister (unsigned long ulBase)
unsigned long QEIPositionGet (unsigned long ulBase)
void QEIPositionSet (unsigned long ulBase, unsigned long ulPosition)
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)
```

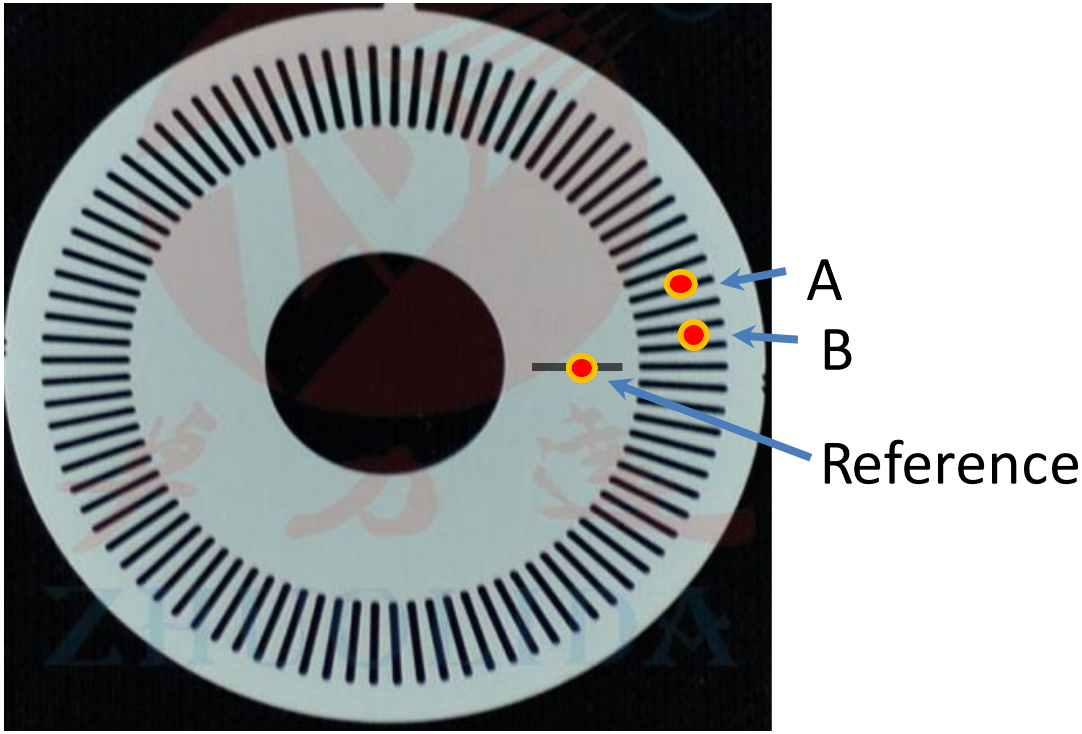
Remember that you are not using the Tiva QEI units in the Project



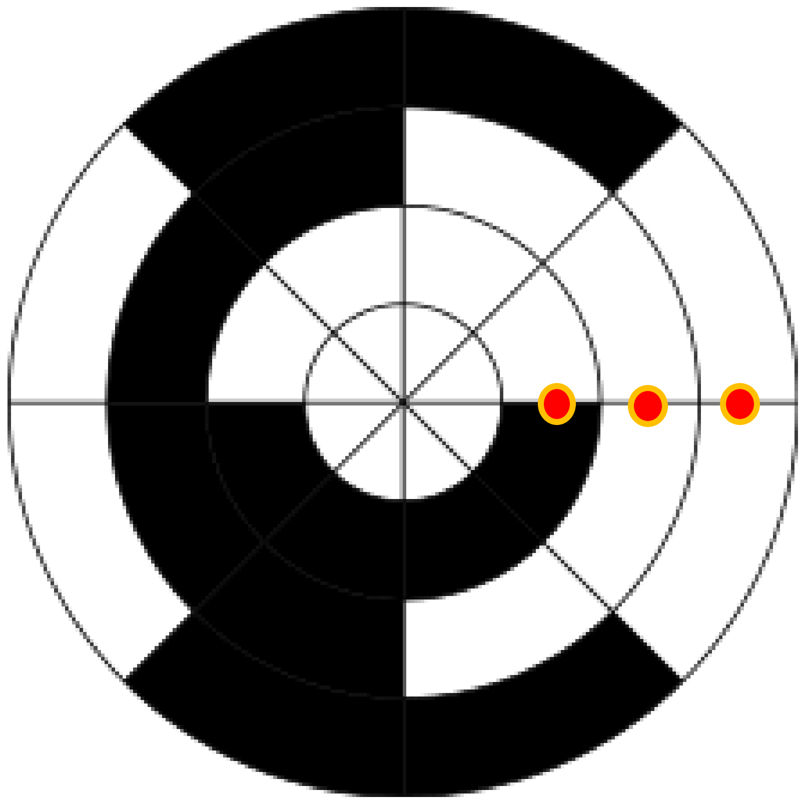
This system only gives us a *relative* location

How can an absolute position be obtained?

Absolute position



Absolute position - alternative



For example: 3-bit binary gray code encoder

Homework

1. Label the transitions between states on the state transition diagram on Slide 6.

2. Why are there no transitions shown between opposing states, i.e. between 1 and 3, or between 2 and 4 on Slide 6?

3. Draw a timing diagram for signals A and B corresponding to clockwise rotation at a *constant rate* followed by counterclockwise rotation at the same rate. Indicate on the diagram the state transitions.

4. If a quadrature encoded disk/sensor unit of the type shown on Slides 2 to 4 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?

5. If a quadrature encoded disk/sensor unit of the type shown on Slides 2 to 4 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?