I^2C Sensor Interface

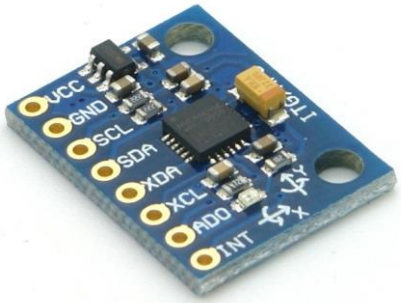
V Muthukumar

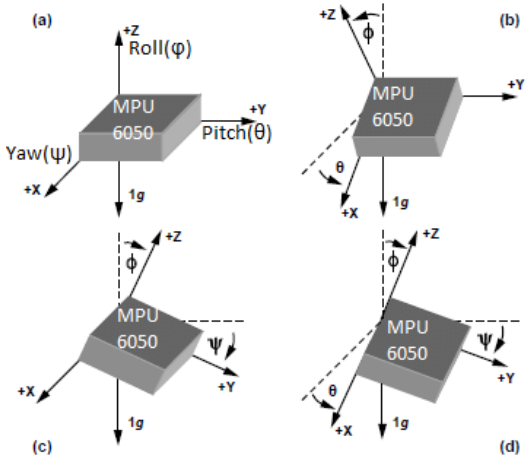
Outline

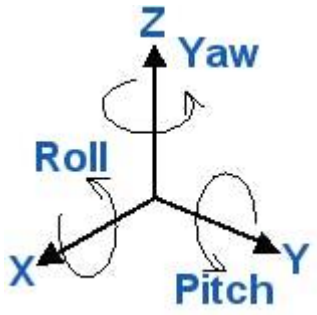
- In this module, the students will learn the following
  - Interface a specific I2C sensors to ATMega328p

MPU-6050

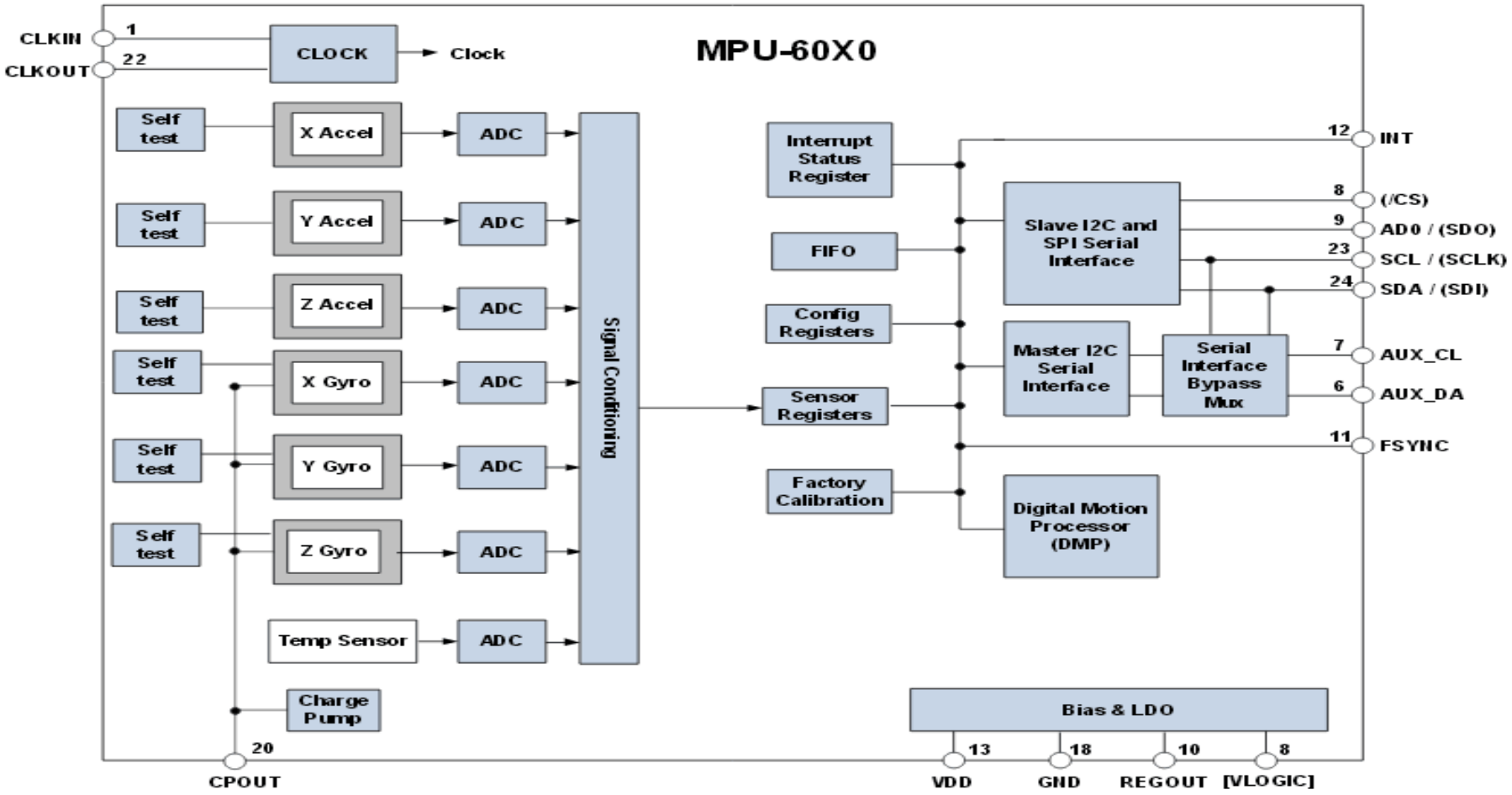
- **GY-521 MPU6050 3-Axis Acceleration Gyroscope 6DOF Module**
- MPU6050 contains both a 3-Axis Gyroscope and a 3-Axis accelerometer allowing measurements of both independently, but all based around the same axes.
- Accelerometer ranges: ±2, ±4, ±8, ±16g
- Gyroscope ranges: ± 250, 500, 1000, 2000 °/s
- Voltage range: 3.3V - 5V (voltage regulator)







MPU-6050


MPU-60X0
CLKIN
CLKOUT
1
22
CLOCK
Clock
Self test
X Accel
Y Accel
Z Accel
X Gyro
Y Gyro
Z Gyro
ADC
Temp Sensor
Charge Pump
20
CPOUT
Signal Conditioning
Interrupt Status Register
FIFO
Config Registers
Sensor Registers
Factory Calibration
Slave I2C and SPI Serial Interface
Master I2C Serial Interface
Serial Interface Bypass Mux
Digital Motion Processor (DMP)
12
INT
8
(/CS)
9
AD0 / (SDO)
23
SCL / (SCLK)
24
SDA / (SDI)
7
AUX_CL
6
AUX_DA
11
FSYNC
Bias & LDO
13
VDD
18
GND
10
REGOUT
8
[VLOGIC]

MPU-6050 Register Map

4.28 Register 107 – Power Management 1
PWR_MGMT_1

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
6B	107	DEVICE _RESET	SLEEP	CYCLE	-	TEMP_DIS	CLKSEL[2:0]		

Description:

This register allows the user to configure the power mode and clock source. It also provides a bit for resetting the entire device, and a bit for disabling the temperature sensor.

4.5 Register 28 – Accelerometer Configuration
ACCEL_CONFIG

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1C	28	XA_ST	YA_ST	ZA_ST	AFS_SEL[1:0]		-		

AFS_SEL=0		16,384
AFS_SEL=1		8,192
AFS_SEL=2		4,096
AFS_SEL=3		2,048

6.2 Accelerometer Specifications

VDD = 2.375V-3.46V, VLOGIC (MPU-6050 only) = 1.8V±5% or VDD, T_A = 25°C

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
ACCELEROMETER SENSITIVITY						
Full-Scale Range	AFS_SEL=0		±2		*g*	
	AFS_SEL=1		±4		*g*	
	AFS_SEL=2		±8		*g*	
	AFS_SEL=3		±16		*g*	
ADC Word Length	Output in two's complement format		16		bits	
Sensitivity Scale Factor	AFS_SEL=0		16,384		LSB/*g*	
	AFS_SEL=1		8,192		LSB/*g*	
	AFS_SEL=2		4,096		LSB/*g*	
	AFS_SEL=3		2,048		LSB/*g*	
Initial Calibration Tolerance			±3		%	
Sensitivity Change vs. Temperature	AFS_SEL=0, -40°C to +85°C		±0.02		%/°C	
Nonlinearity	Best Fit Straight Line		0.5		%	
Cross-Axis Sensitivity			±2		%	

MPU-6050 Register Map

4.4 Register 27 – Gyroscope Configuration
GYRO_CONFIG

Type: Read/Write

Register (Hex)	Register (Decimal)	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1B	27	XG_ST	YG_ST	ZG_ST	FS_SEL[1:0]		-	-	-

FS_SEL selects the full scale range of the gyroscope outputs according to the following table.

FS_SEL	Full Scale Range
0	± 250 °/s
1	± 500 °/s
2	± 1000 °/s
3	± 2000 °/s

6.1 Gyroscope Specifications

VDD = 2.375V-3.46V, VLOGIC (MPU-6050 only) = 1.8V±5% or VDD, T_A = 25°C

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
GYROSCOPE SENSITIVITY						
Full-Scale Range	FS_SEL=0		±250		°/s	
	FS_SEL=1		±500		°/s	
	FS_SEL=2		±1000		°/s	
	FS_SEL=3		±2000		°/s	
Gyroscope ADC Word Length			16		bits	
Sensitivity Scale Factor	FS_SEL=0		131		LSB/(°/s)	
	FS_SEL=1		65.5		LSB/(°/s)	
	FS_SEL=2		32.8		LSB/(°/s)	
	FS_SEL=3		16.4		LSB/(°/s)	
Sensitivity Scale Factor Tolerance	25°C	-3		+3	%	
Sensitivity Scale Factor Variation Over Temperature			±2		%	
Nonlinearity	Best fit straight line; 25°C		0.2		%	
Cross-Axis Sensitivity			±2		%	

MPU-6050 Initialization

```
void MPU6050_Init()/* Gyro initialization function */
{
  _delay_ms(150);/* Power up time >100ms */
  I2C_Start_Wait(0xD0);/* Start with device write address */
  I2C_Write(SMPLRT_DIV);/* Write to sample rate register */
  I2C_Write(0x07);/* 1KHz sample rate */
  I2C_Stop();

  I2C_Start_Wait(0xD0);
  I2C_Write(PWR_MGMT_1);/* Write to power management register */
  I2C_Write(0x01);/* X axis gyroscope reference frequency */
  I2C_Stop();

  I2C_Start_Wait(0xD0);
  I2C_Write(CONFIG);/* Write to Configuration register */
  I2C_Write(0x00);/* Fs = 8KHz */
  I2C_Stop();

  I2C_Start_Wait(0xD0);
  I2C_Write(GYRO_CONFIG);/* Write to Gyro configuration register */
  I2C_Write(0x18);/* Full scale range +/- 2000 degree/C */
  I2C_Stop();

  I2C_Start_Wait(0xD0);
  I2C_Write(INT_ENABLE);/* Write to interrupt enable register */
  I2C_Write(0x01);
  I2C_Stop();
}
```

MPU-6050 Individual Read & Write Function

```
void MPU6050_writereg(uint8_t reg, uint8_t val)
{
i2c_start(MPU6050+I2C_WRITE);
i2c_write(reg);  // go to register e.g. 106 user control
i2c_write(val);  // set value e.g. to 0100 0000 FIFO enable
i2c_stop();      // set stop condition = release bus
}

uint16_t MPU6050_readreg(uint8_t reg)
{
i2c_start_wait(MPU6050+I2C_WRITE);  // set device address and write mode
i2c_write(reg);                     // ACCEL_XOUT
i2c_rep_start(MPU6050+I2C_READ);    // set device address and read mode
raw = i2c_readAck();                // read one intermediate byte
raw = (raw<<8) | i2c_readNak();     // read last byte
i2c_stop();

return raw;
}
```

```
// go to register 107 set value to 0000 0000 and wake up sensor
MPU6050_writereg(0x6B, 0x00);


 // read raw X acceleration from fifo
 Acc_x = MPU6050_readreg(0x3B);
```

MPU-6050: Accelerometer Register Map

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
3B	59	ACCEL_XOUT_H	R	ACCEL_XOUT[15:8]							
3C	60	ACCEL_XOUT_L	R	ACCEL_XOUT[7:0]							
3D	61	ACCEL_YOUT_H	R	ACCEL_YOUT[15:8]							
3E	62	ACCEL_YOUT_L	R	ACCEL_YOUT[7:0]							
3F	63	ACCEL_ZOUT_H	R	ACCEL_ZOUT[15:8]							
40	64	ACCEL_ZOUT_L	R	ACCEL_ZOUT[7:0]							
41	65	TEMP_OUT_H	R	TEMP_OUT[15:8]							
42	66	TEMP_OUT_L	R	TEMP_OUT[7:0]							

MPU6050 – Gyroscope Register Map

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
43	67	GYRO_XOUT_H	R	GYRO_XOUT[15:8]							
44	68	GYRO_XOUT_L	R	GYRO_XOUT[7:0]							
45	69	GYRO_YOUT_H	R	GYRO_YOUT[15:8]							
46	70	GYRO_YOUT_L	R	GYRO_YOUT[7:0]							
47	71	GYRO_ZOUT_H	R	GYRO_ZOUT[15:8]							
48	72	GYRO_ZOUT_L	R	GYRO_ZOUT[7:0]							

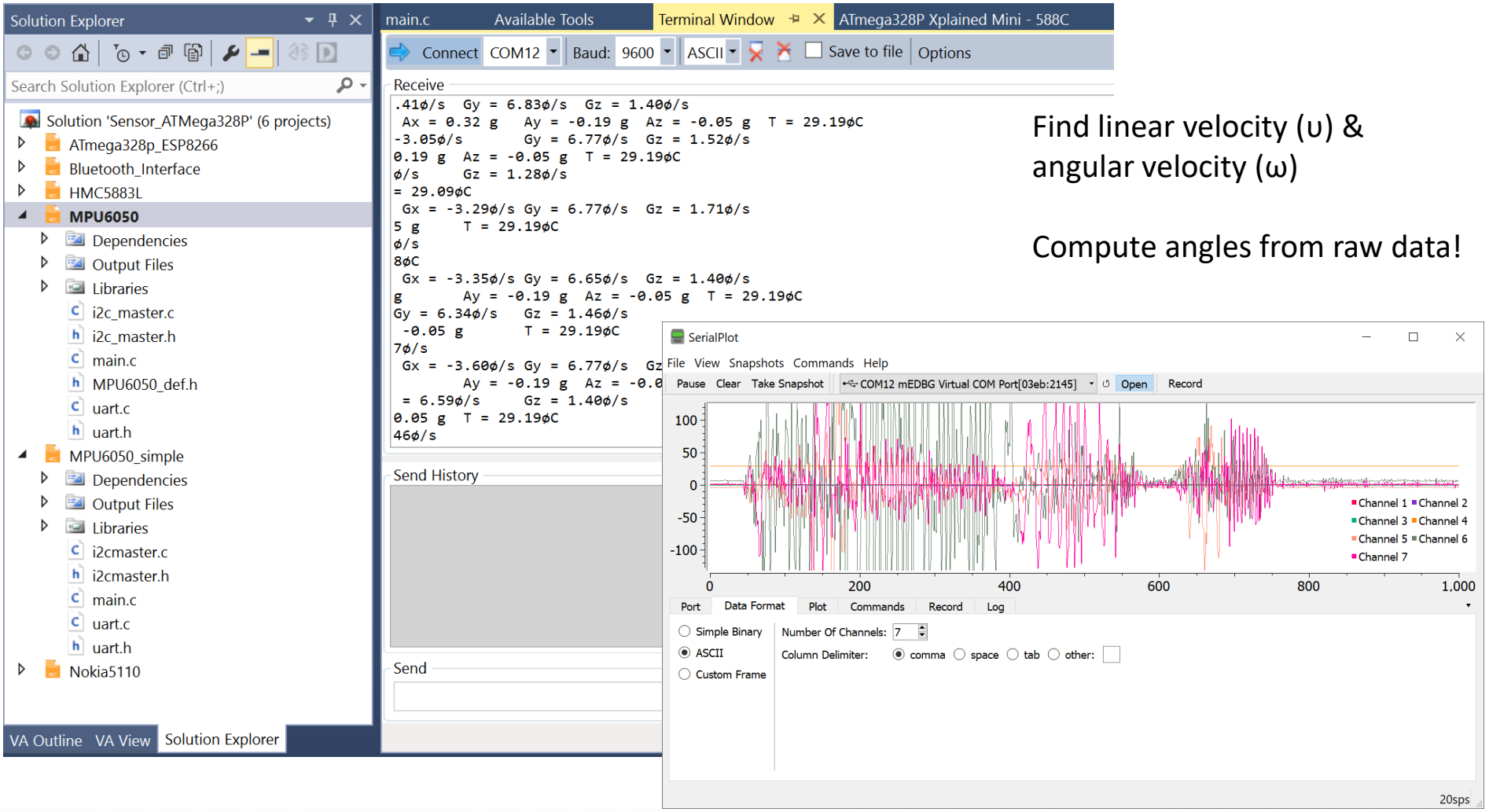
MPU-6050 Continuous Read & Write Function

```
void MPU_Start_Loc()
{
  I2C_Start_Wait(0xD0);/* I2C start with device write address */
  I2C_Write(ACCEL_XOUT_H);/* Write start location address from where to read */
  I2C_Repeated_Start(0xD1);/* I2C start with device read address */
}

void Read_RawValue()
{
  MPU_Start_Loc();/* Read Gyro values */
  Acc_x = (((int)I2C_Read_Ack()<<8) | (int)I2C_Read_Ack());
  //.. Read other registers
  I2C_Stop();
}
```

MPU-6050 Main Function

```
                    #include "MPU6050_def.h"/* Include MPU6050 register define file */
                    #include "i2c_master.h"/* Include I2C Master header file */
                    #include "uart.h"/* Include USART header file */
int main()
{
  char buffer[20], float_[10];
  float Xa;
  I2C_Init();/* Initialize I2C */
  MPU6050_Init();/* Initialize MPU6050 */
  USART_Init(9600);/* Initialize USART with 9600 baud rate */

  while(1)
  {
    Read_RawValue();
    /* Divide raw value by sensitivity scale factor to get real values */
    Xa = Acc_x/16384.0;
    /* Take values in buffer to send all parameters over USART */
    dtostrf( Xa, 3, 2, float_ );
    sprintf(buffer," Ax = %s g\t",float_);
    USART_SendString(buffer);
    …
  }
}
```

MPU-6050 Demo



Find linear velocity (υ) & angular velocity (ω)

Compute angles from raw data!

Summary

- On completion of this module student should be able to
  - Understand I^2C module in AVRs
  - Program using assembly and C program to store and retrieve data in/from an I^2C device