TSLPB V3 Library 0.1.0

Generated by Doxygen 1.8.14

Contents

1 Twiggs Space Lab Payload Board Driver							
	1.1	Usage		1			
		1.1.1	Example	1			
2	Clas	s Index		3			
	2.1	Class I	st	3			
3	File	Index		5			
	3.1	File Lis		5			
4	Clas	s Docu	entation	7			
	4.1	Thinsa	Packet_t Union Reference	7			
		4.1.1	Detailed Description	7			
		4.1.2	Member Data Documentation	7			
			4.1.2.1 NSLPacket	7			
			4.1.2.2 payloadData	8			
	4.2	TSLPE	Class Reference	8			
		4.2.1	Detailed Description	8			
		4.2.2	Member Function Documentation	8			
			4.2.2.1 begin()	9			
			4.2.2.2 isClearToSend()	9			
			4.2.2.3 pushDataToNSL()	9			
			4.2.2.4 read8bitRegister()	10			
			4.2.2.5 readAnalogSensor()	10			
			4.2.2.6 readDigitalSensor()	10			

ii CONTENTS

		4.2.2.7	readDigitalSensorRaw()	11
		4.2.2.8	sleepUntilClearToSend()	11
4.3	UserDa	ataStruct_	t Struct Reference	11
	4.3.1	Detailed	Description	12
	4.3.2	Member	Data Documentation	13
		4.3.2.1	bmePres	13
		4.3.2.2	bmeTemp	13
		4.3.2.3	bnoCal	13
		4.3.2.4	bnomagx	13
		4.3.2.5	bnomagy	13
		4.3.2.6	bnomagz	14
		4.3.2.7	header	14
		4.3.2.8	quatw	14
		4.3.2.9	quatx	14
		4.3.2.10	quaty	14
		4.3.2.11	quatz	14
		4.3.2.12	tslCurrent	15
		4.3.2.13	tslMagXraw	15
		4.3.2.14	tslMagYraw	15
		4.3.2.15	tslMagZraw	15
		4.3.2.16	tslTempExt	15
		4.3.2.17	tslVolts	15
		4.3.2.18	unused5	15

CONTENTS

5	File	Docume	mentation 1				
	5.1	MPU92	250_REGS	S.h File Reference	17		
		5.1.1	Detailed Description				
		5.1.2	Macro Definition Documentation				
			5.1.2.1	ACC_FULL_SCALE_16_G	19		
			5.1.2.2	ACC_FULL_SCALE_2_G	19		
			5.1.2.3	ACC_FULL_SCALE_4_G	19		
			5.1.2.4	ACC_FULL_SCALE_8_G	19		
			5.1.2.5	GYRO_FULL_SCALE_1000_DPS	19		
			5.1.2.6	GYRO_FULL_SCALE_2000_DPS	19		
			5.1.2.7	GYRO_FULL_SCALE_250_DPS	20		
			5.1.2.8	GYRO_FULL_SCALE_500_DPS	20		
			5.1.2.9	MAG_MAX_BYTE_VALUE	20		
			5.1.2.10	MAG_MAX_VALUE_FLOAT	20		
		5.1.3	Enumera	tion Type Documentation	20		
			5.1.3.1	anonymous enum	20		
			5.1.3.2	MPU9250_GYRO_REGISTER_t	21		
			5.1.3.3	MPU9250_MAG_CONTROL_t	21		
			5.1.3.4	MPU9250_MAG_REGISTER_t	22		
			5.1.3.5	MPU9250_TEMP_REGISTER_t	22		
		5.1.4	Variable I	Documentation	22		
			5.1.4.1	MPU9250_ACCEL_REGISTER_t	22		
	5.2	NSL_T	hinSat.h F	ile Reference	23		
		5.2.1	Detailed	Description	23		
		5.2.2	Macro De	efinition Documentation	23		
			5.2.2.1	NSL_BAUD_RATE	23		
			5.2.2.2	NSL_PACKET_HEADER	24		
			5.2.2.3	NSL_PACKET_HEADER_LENGTH	24		
			5.2.2.4	NSL_PACKET_SIZE	24		
			5.2.2.5	NSL_SERIAL_ACK	24		

iv CONTENTS

		5.2.2.6	NSL_SERIAL_BUSY	24
		5.2.2.7	NSL_SERIAL_NAK	24
		5.2.2.8	NSL_SERIAL_READY	24
5.3	ThinSa	it_DataPac	ket.h File Reference	25
	5.3.1	Detailed	Description	25
5.4	TSLPE	3.cpp File F	Reference	25
	5.4.1	Detailed	Description	25
5.5	TSLPE	3.h File Ref	erence	26
	5.5.1	Detailed	Description	27
	5.5.2	Macro De	efinition Documentation	27
		5.5.2.1	LMA_TEMP_REG_DEGREES_PER_LSB	27
		5.5.2.2	LMA_TEMP_REG_SIGN_BIT	28
		5.5.2.3	LMA_TEMP_REG_UNUSED_LSBS	28
		5.5.2.4	TSL_ADC	28
		5.5.2.5	TSL_MUX_A	28
		5.5.2.6	TSL_MUX_B	28
		5.5.2.7	TSL_MUX_C	28
		5.5.2.8	TSL_MUX_RESPONSE_TIME	29
		5.5.2.9	TSL_SENSOR_READY_TIMEOUT	29
		5.5.2.10	TSL_SERIAL_STATUS_PIN	29
	5.5.3	Enumera	tion Type Documentation	29
		5.5.3.1	LM75A_REG	29
		5.5.3.2	TSLPB_AnalogSensor_t	29
		5.5.3.3	TSLPB_DigitalSensor_t	30
		5.5.3.4	TSLPB_I2CAddress_t	30
Index				33

Chapter 1

Twiggs Space Lab Payload Board Driver

TSLPB is a driver class that can be instantiated and used to access the sensors and devices on the TSLPB V3 for the ThinSat program.

The driver sets up all the input and output pins required for accessing the analog sensors, and provides methods for reading both the analog and digital sensors.

1.1 Usage

You will need to do the following to use this library:

- 1. Include tslbp.h in your program.
- 2. Instantiate a TSLPB object
- 3. Run the TSLPB::begin() method

Once these steps are complete, you may call any of the public methods to interact with the TSL Payload Board.

1.1.1 Example

```
#include "TSLPB.h"

TSLBP tslpb;

void setup() {
    tslpb.begin();
}

void loop() {
    uint16_t tslVolts = tslpb.readAnalogSensor(Voltage);
    uint16_t tslCurrent = tslpb.readAnalogSensor(Current);
    uint16_t tslTempExt = tslpb.readAnalogSensor(TempExt);

    uint16_t tslDT1Raw = tslpb.readTslDigitalSensorRaw(DT1);
    double tslDT1C = tslpb.readTslDigitalSensor(DT1);
}
```

You probably noticed the "Voltage", "Current", etc arguments. The TSLPB driver has two enums that allow the client to call the read methods with human-readable code, and without worrying about keeping I2C addresses or managing low-level mux switching.

- TSLPB_AnalogSensor_t
- · TSLPB DigitalSensor t

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ThinsatF	Packet_t	
	A union of the UserDataStruct_t payloadData and a byte array that is used to send the user's mission data to the NSL Mothership	7
TSLPB		
	The controller class for the TSL Payload Board. Create an instance of this class to use its member functions for accessing the onboard analog and digital sensors. Methods for communicating with the NSL Mothership are also included	8
UserDat	taStruct_t	
	A user-customizable structure to hold any data the user intends to send back to Earth	11

4 Class Index

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

MPU9250_REGS.h	
Register map and configuration data for the MPU9250 IMU on the TSLPB V3	17
NSL_ThinSat.h	
Function prototypes, includes, and definitions for NSL to TSLPB Arduino interface	23
ThinSat_DataPacket.h	
Defines the custom data structure used to store the user's payload data, and the union that is	
used to transmit the data to the NSL Mothership	25
TSLPB.cpp	
Implementation of TSLPB interface for Arduino	25
TSLPB.h	
Function prototypes, includes, and definitions for TSLPB Arduino interface	26

6 File Index

Chapter 4

Class Documentation

4.1 ThinsatPacket_t Union Reference

A union of the UserDataStruct_t payloadData and a byte array that is used to send the user's mission data to the NSL Mothership.

#include <ThinSat_DataPacket.h>

Public Attributes

- UserDataStruct_t payloadData
- byte NSLPacket [sizeof(UserDataStruct_t)]

4.1.1 Detailed Description

A union of the <u>UserDataStruct_t</u> payloadData and a byte array that is used to send the user's mission data to the NSL Mothership.

Warning

DO NOT MODIFY THIS UNION UNLESS YOU REALLY REALLY KNOW WHAT YOU ARE DOING. This datatype is used in the public method TSLPB::pushDataToNSL(ThinsatPacket_t data) and changing this union may break that functionality.

4.1.2 Member Data Documentation

4.1.2.1 NSLPacket

byte ThinsatPacket_t::NSLPacket[sizeof(UserDataStruct_t)]

8 Class Documentation

4.1.2.2 payloadData

UserDataStruct_t ThinsatPacket_t::payloadData

The documentation for this union was generated from the following file:

ThinSat_DataPacket.h

4.2 TSLPB Class Reference

The controller class for the TSL Payload Board. Create an instance of this class to use its member functions for accessing the onboard analog and digital sensors. Methods for communicating with the NSL Mothership are also included.

#include <TSLPB.h>

Public Member Functions

· void begin ()

Initializes the TSLPB, starts the I2C bus, and configures the pins needed for reading the TSLPB analog sensors.

uint8_t readAnalogSensor (TSLPB_AnalogSensor_t sensorName)

This method returns the raw value from the specified analog sensor.

double readDigitalSensor (TSLPB_DigitalSensor_t sensor)

This API returns the process from the specified sensor as a double-precision floating point value in the appropriate units for the sensor.

uint16_t readDigitalSensorRaw (TSLPB_DigitalSensor_t sensor)

This API returns the raw value from the specified sensor. Handles endiannes and discarding unused bits.

- void sleepUntilClearToSend ()
- bool isClearToSend ()

This function returns true if the NSL Mothership is ready to receive data over the serial line.

bool pushDataToNSL (ThinsatPacket_t data)

This function sends the user's payload data to NSL Mothership over the serial line. This function expects a ThinsatPacket_t union as an argument. That data type is defined in ThinSat_DataPacket.h, and the contents of the user data structure may be customized.

uint8_t read8bitRegister (TSLPB_I2CAddress_t i2cAddress, const uint8_t reg)

This private method returns the contents of a single I2C register (1 byte)

4.2.1 Detailed Description

The controller class for the TSL Payload Board. Create an instance of this class to use its member functions for accessing the onboard analog and digital sensors. Methods for communicating with the NSL Mothership are also included.

4.2.2 Member Function Documentation

4.2 TSLPB Class Reference 9

4.2.2.1 begin()

```
void TSLPB::begin ( )
```

Initializes the TSLPB, starts the I2C bus, and configures the pins needed for reading the TSLPB analog sensors.

Call this function in the setup() function as follows:

```
void setup() {
    tslpb.begin();
}
```

Note

This function changes the state of 4 I/O pins:

PIN	MODE
TSL_ADC	Analog Input
TSL_MUX_A	Digital Output
TSL_MUX_B	Digital Output
TSL_MUX_C	Digital Output
TSL_SERIAL_STATUS_PIN	Digital Input

4.2.2.2 isClearToSend()

```
bool TSLPB::isClearToSend ( )
```

This function returns true if the NSL Mothership is ready to receive data over the serial line.

Returns

true or false

4.2.2.3 pushDataToNSL()

This function sends the user's payload data to NSL Mothership over the serial line. This function expects a ThinsatPacket_t union as an argument. That data type is defined in ThinSat_DataPacket.h, and the contents of the user data structure may be customized.

Parameters

ii	l	data	A ThinsatPacket_t union.
----	---	------	--------------------------

10 Class Documentation

Returns

nominal transmission: true or false

4.2.2.4 read8bitRegister()

This private method returns the contents of a single I2C register (1 byte)

Parameters

in	i2cAddress	TSLPB Digital Sensor Address Enum (a uint8_t I2C address)
in	reg	Register (a uint8_t I2C register)

Returns

uint8_t The contents of register reg at I2C device i2cAddress

4.2.2.5 readAnalogSensor()

This method returns the raw value from the specified analog sensor.

Parameters

in sensorName : TSLPB_AnalogSensor_t Sensor Nam	enum
---	------

Returns

a uint8_t containing raw value of the Arduino Pro Mini's ADC.

4.2.2.6 readDigitalSensor()

This API returns the process from the specified sensor as a double-precision floating point value in the appropriate units for the sensor.

Parameters

	in	sensorName	TSLPB_DigitalSensor_t Sensor Name Selection Enum]
--	----	------------	--	---

Returns

a value in the appropriate units for the sensor as a double precision floating point value.

4.2.2.7 readDigitalSensorRaw()

This API returns the raw value from the specified sensor. Handles endiannes and discarding unused bits.

Parameters

	in	sensorName	: TSLPB_DigitalSensor_t Sensor Name Enum	
--	----	------------	--	--

Returns

a uint16_t containing the bit pattern from the sensor's register.

- < I2C buffer for read function
- < return value, after endian correction

4.2.2.8 sleepUntilClearToSend()

```
void TSLPB::sleepUntilClearToSend ( )
```

The documentation for this class was generated from the following files:

- TSLPB.h
- TSLPB.cpp

4.3 UserDataStruct_t Struct Reference

A user-customizable structure to hold any data the user intends to send back to Earth.

```
#include <ThinSat_DataPacket.h>
```

12 **Class Documentation**

```
Public Attributes

    char header [NSL_PACKET_HEADER_LENGTH]

    int16_t quatw

          1 - 3 (value from -1000 to 1000) 1.000 (unitless)
    • int16 t quatx
          4 - 5 (value from -1000 to 1000) 1.000 (unitless)
    · int16 t quaty
          6 - 7 (value from -1000 to 1000) 1.000 (unitless)
    • int16_t quatz
          8 - 9 (value from -1000 to 1000) 1.000 (unitless)
    · int16_t bnomagx
          10 - 11 (value from -20480 to 20470) 2047.0 uT (from BNO)
    · int16_t bnomagy
          12 - 13 (value from -20480 to 20470) 2047.0 uT
    · int16_t bnomagz
          14 - 15 (value from -20480 to 20470) 2047.0 uT

    uint8_t bnoCal

          16 (sys, gyro, accel, mag) 01010101b

    unsigned long bmePres

          17 - 20 (values from 0 to 1010000) 101000.0 Pa

    int16_t bmeTemp

          21 - 22 (values from -1000 to 1000) 100.0 C

    uint16_t tslTempExt

          23 - 24 (10 bits 0-1023) ADC Raw Counts

    uint16_t tslVolts

          25 - 26 (10 bits 0-1023) ADC Raw Counts
    • uint16 ttslCurrent
```

- 27 28 (10 bits 0-1023) ADC Raw Counts
- int16_t tslMagXraw
 - 29 30 Raw value (2's compliment form) -0x7FF8 to 0x7FF8
- int16 t tslMagYraw
 - 31 32 Raw value (2's compliment form) -0x7FF8 to 0x7FF8
- int16_t tslMagZraw
 - 33 34 Raw value (2's compliment form) -0x7FF8 to 0x7FF8
- uint8 t unused5
 - 35 Left over for status bits?

4.3.1 **Detailed Description**

A user-customizable structure to hold any data the user intends to send back to Earth.

Note

This is a sample UserDataStruct_t. It was developed for the VCSFA ThinSat custom payload. Some of the fields are for external sensors and some of the fields are for TSLPB sensors.

We recommend adding comments that show the expected ranges and units of any data being put into a field. This will ensure that you can translate the data later.

Warning

The struct must be NSL_PACKET_SIZE bytes in total size. The first member must always be called "header" and have a size of NSL_PACKET_HEADER_LENGTH

4.3.2 Member Data Documentation

4.3.2.1 bmePres unsigned long UserDataStruct_t::bmePres 17 - 20 (values from 0 to 1010000) 101000.0 Pa 4.3.2.2 bmeTemp int16_t UserDataStruct_t::bmeTemp 21 - 22 (values from -1000 to 1000) 100.0 C 4.3.2.3 bnoCal uint8_t UserDataStruct_t::bnoCal 16 (sys, gyro, accel, mag) 01010101b 4.3.2.4 bnomagx int16_t UserDataStruct_t::bnomagx 10 - 11 (value from -20480 to 20470) 2047.0 uT (from BNO) 4.3.2.5 bnomagy

Generated by Doxygen

int16_t UserDataStruct_t::bnomagy

12 - 13 (value from -20480 to 20470) 2047.0 uT

14 Class Documentation

```
4.3.2.6 bnomagz
int16_t UserDataStruct_t::bnomagz
14 - 15 (value from -20480 to 20470) 2047.0 uT
4.3.2.7 header
char UserDataStruct_t::header[NSL_PACKET_HEADER_LENGTH]
4.3.2.8 quatw
int16_t UserDataStruct_t::quatw
1 - 3 (value from -1000 to 1000) 1.000 (unitless)
4.3.2.9 quatx
int16_t UserDataStruct_t::quatx
4 - 5 (value from -1000 to 1000) 1.000 (unitless)
4.3.2.10 quaty
int16_t UserDataStruct_t::quaty
6 - 7 (value from -1000 to 1000) 1.000 (unitless)
4.3.2.11 quatz
int16_t UserDataStruct_t::quatz
8 - 9 (value from -1000 to 1000) 1.000 (unitless)
```

4.3.2.12 tslCurrent

uint16_t UserDataStruct_t::tslCurrent

27 - 28 (10 bits 0-1023) ADC Raw Counts

4.3.2.13 tslMagXraw

int16_t UserDataStruct_t::tslMagXraw

29 - 30 Raw value (2's compliment form) -0x7FF8 to 0x7FF8

4.3.2.14 tslMagYraw

int16_t UserDataStruct_t::tslMagYraw

31 - 32 Raw value (2's compliment form) -0x7FF8 to 0x7FF8

4.3.2.15 tslMagZraw

int16_t UserDataStruct_t::tslMagZraw

33 - 34 Raw value (2's compliment form) -0x7FF8 to 0x7FF8

4.3.2.16 tslTempExt

uint16_t UserDataStruct_t::tslTempExt

23 - 24 (10 bits 0-1023) ADC Raw Counts

4.3.2.17 tslVolts

uint16_t UserDataStruct_t::tslVolts

25 - 26 (10 bits 0-1023) ADC Raw Counts

4.3.2.18 unused5

uint8_t UserDataStruct_t::unused5

35 Left over for status bits?

The documentation for this struct was generated from the following file:

ThinSat_DataPacket.h

16 Class Documentation

Chapter 5

File Documentation

5.1 MPU9250_REGS.h File Reference

Register map and configuration data for the MPU9250 IMU on the TSLPB V3.

Macros

- #define GYRO_FULL_SCALE_250_DPS 0x00
- #define GYRO_FULL_SCALE_500_DPS 0x08

Gyroscope range parameter.

Gyroscope range parameter.

#define GYRO_FULL_SCALE_1000_DPS 0x10

Gyroscope range parameter.

• #define GYRO_FULL_SCALE_2000_DPS 0x18

Gyroscope range parameter.

• #define ACC_FULL_SCALE_2_G 0x00

Accelerometer range parameter.

• #define ACC_FULL_SCALE_4_G 0x08

Accelerometer range parameter.

• #define ACC_FULL_SCALE_8_G 0x10

Accelerometer range parameter.

• #define ACC_FULL_SCALE_16_G 0x18

Accelerometer range parameter.

• #define MAG_MAX_BYTE_VALUE 0x7FF8

Max register for scaling.

• #define MAG_MAX_VALUE_FLOAT 4912

in units of uT for scaling

Enumerations

```
enum MPU9250_TEMP_REGISTER_t { MPU9250_TEMP_OUT_MSB = 0x41, MPU9250_TEMP_OUT_LSB
 = 0x42
• enum {
 MPU9250_ACCEL_XOUT_MSB = 0x3B, MPU9250_ACCEL_XOUT_LSB = 0x3C, MPU9250_ACCEL_YOUT_MSB
 = 0x3D, MPU9250_ACCEL_YOUT_LSB = 0x3E,
 MPU9250 ACCEL ZOUT MSB = 0x3F, MPU9250 ACCEL ZOUT LSB = 0x40, MPU9250 ACCEL SELF TEST X
 = 0x0D, MPU9250 ACCEL SELF TEST Y = 0x0E,
 MPU9250 ACCEL SELF TEST Z = 0x0F }

    enum MPU9250 GYRO REGISTER t {

 MPU9250 GYRO XOUT MSB = 0x43, MPU9250 GYRO XOUT LSB = 0x44, MPU9250 GYRO YOUT MSB
 = 0x45, MPU9250 GYRO YOUT LSB = 0x46,
 MPU9250 GYRO ZOUT MSB = 0x47, MPU9250 GYRO ZOUT LSB = 0x48, MPU9250 GYRO SELF TEST X
 = 0x00, MPU9250 GYRO SELF TEST Y = 0x01,
 MPU9250_GYRO_SELF_TEST_Z = 0x02 }
• enum MPU9250 MAG REGISTER t {
 MPU9250 MAG REG DEVICE ID = 0x00, MPU9250 MAG REG INFORMATION = 0x01, MPU9250 MAG REG STATUS
 = 0x02, MPU9250_MAG_REG_X_DATA_LSB = 0x03,
 MPU9250 MAG REG X DATA MSB = 0x04, MPU9250 MAG REG Y DATA LSB = 0x05, MPU9250 MAG REG Y DATA
 = 0x06, MPU9250 MAG REG Z DATA LSB = 0x07,
 MPU9250 MAG_REG_Z_DATA_MSB = 0x08, MPU9250_MAG_REG_STATUS_2 = 0x09, MPU9250_MAG_REG_CONTROL
 = 0x0A, MPU9250_MAG_REG_SELF_TEST = 0x0C,
 MPU9250_MAG_REG_I2C_DISABLE = 0x0F, MPU9250_MAG_REG_X_SENSITIVITY = 0x10, MPU9250_MAG_REG_Y_SEN
 = 0x11, MPU9250 MAG REG Z SENSITIVITY = 0x12 }

    enum MPU9250 MAG CONTROL t {

 MPU9250 MAG STATUS 1 DATA READY BIT = 0x00, MPU9250 REG INT PIN BYPASS = 0x37,
 MPU9250 PASSTHROUGH ON = 0x02, MPU9250 PASSTHROUGH OFF = 0x00,
 MAG MODE SINGLE MEAS = 0b0001, MAG MODE CONTINUOUS 8HZ = 0b0010, MAG MODE CONTINUOUS 100HZ
 = 0b0011, MAG MODE POWER DOWN = 0b0000,
 MAG MODE SELF TEST = 0b1000, MAG MODE BITMASK = 0x0F, MAG MODE 14 BIT = 0x00,
 MAG_MODE_16_BIT = 0x10,
 MAG MASK DATA OVERRUN = 0x02, MAG MASK DATA READY = 0x01, MAG MASK DATA OVERFLOW
 = 0x08, MAG_MASK_DATA_BIT_RESOLUTION = 0x10 }
```

Variables

• enum { ... } MPU9250 ACCEL REGISTER t

5.1.1 Detailed Description

Register map and configuration data for the MPU9250 IMU on the TSLPB V3.

Author

Nicholas Counts

Date

06/19/18

5.1.2 Macro Definition Documentation

5.1.2.1 ACC_FULL_SCALE_16_G

#define ACC_FULL_SCALE_16_G 0x18

Accelerometer range parameter.

5.1.2.2 ACC_FULL_SCALE_2_G

#define ACC_FULL_SCALE_2_G 0x00

Accelerometer range parameter.

5.1.2.3 ACC_FULL_SCALE_4_G

#define ACC_FULL_SCALE_4_G 0x08

Accelerometer range parameter.

5.1.2.4 ACC_FULL_SCALE_8_G

#define ACC_FULL_SCALE_8_G 0x10

Accelerometer range parameter.

5.1.2.5 GYRO_FULL_SCALE_1000_DPS

#define GYRO_FULL_SCALE_1000_DPS 0x10

Gyroscope range parameter.

5.1.2.6 GYRO_FULL_SCALE_2000_DPS

#define GYRO_FULL_SCALE_2000_DPS 0x18

Gyroscope range parameter.

5.1.2.7 GYRO_FULL_SCALE_250_DPS

#define GYRO_FULL_SCALE_250_DPS 0x00

Gyroscope range parameter.

5.1.2.8 GYRO_FULL_SCALE_500_DPS

#define GYRO_FULL_SCALE_500_DPS 0x08

Gyroscope range parameter.

5.1.2.9 MAG_MAX_BYTE_VALUE

#define MAG_MAX_BYTE_VALUE 0x7FF8

Max register for scaling.

5.1.2.10 MAG_MAX_VALUE_FLOAT

#define MAG_MAX_VALUE_FLOAT 4912

in units of uT for scaling

5.1.3 Enumeration Type Documentation

5.1.3.1 anonymous enum

anonymous enum

Enumerator

MPU9250_ACCEL_XOUT_MSB	
MPU9250_ACCEL_XOUT_LSB	
MPU9250_ACCEL_YOUT_MSB	
MPU9250_ACCEL_YOUT_LSB	
MPU9250_ACCEL_ZOUT_MSB	
MPU9250_ACCEL_ZOUT_LSB	
MPU9250_ACCEL_SELF_TEST↔	
_X	
MPU9250_ACCEL_SELF_TEST↔	
_Y	
MPU9250 ACCEL SELF TEST←	ı

Ζ

5.1.3.2 MPU9250_GYRO_REGISTER_t

enum MPU9250_GYRO_REGISTER_t

Enumerator

MPU9250_GYRO_XOUT_MSB	
MPU9250_GYRO_XOUT_LSB	
MPU9250_GYRO_YOUT_MSB	
MPU9250_GYRO_YOUT_LSB	
MPU9250_GYRO_ZOUT_MSB	
MPU9250_GYRO_ZOUT_LSB	
MPU9250_GYRO_SELF_TEST↔	
_X	
MPU9250_GYRO_SELF_TEST↔	
_Y	
MPU9250_GYRO_SELF_TEST↔	
_Z	

$5.1.3.3 \quad \mathsf{MPU9250_MAG_CONTROL_t}$

enum MPU9250_MAG_CONTROL_t

Enumerator

MPU9250_MAG_STATUS_1_DATA_READY_BIT	
MPU9250_REG_INT_PIN_BYPASS	READ/WRITE: Allow passthrough mode.
MPU9250_PASSTHROUGH_ON	When asserted, the i2c_master interface pins go into 'bypass mode' when the i2c master interface is disabled. The pins will float high due to the internal pull-up if not enabled and the i2c master interface is disabled.
MPU9250_PASSTHROUGH_OFF	& with current register
MAG_MODE_SINGLE_MEAS	Single Measurement Mode.
MAG_MODE_CONTINUOUS_8HZ	Continuous register update mode (8 Hz)
MAG_MODE_CONTINUOUS_100HZ	Continuous register update mode (100 Hz)
MAG_MODE_POWER_DOWN	Low power standby mode.
MAG_MODE_SELF_TEST	Perform a self test with internal magnetic field generator.
MAG_MODE_BITMASK	bit mask for mode-setting register
MAG_MODE_14_BIT	bit 4 off for 14-bit output
MAG_MODE_16_BIT	bit 4 on for 16-bit output
MAG_MASK_DATA_OVERRUN	ST1 bit mask for data overrun.
MAG_MASK_DATA_READY	ST1 bit mask for data ready.
MAG_MASK_DATA_OVERFLOW	ST2 bit mask for "Magnetic sensor overflow occurred" - true if true.
MAG_MASK_DATA_BIT_RESOLUTION	ST2 bit mask: 0 if 14-bit output, 1 if 16-bit output.

Generated by Doxygen

5.1.3.4 MPU9250_MAG_REGISTER_t

enum MPU9250_MAG_REGISTER_t

Enumerator

MPU9250_MAG_REG_DEVICE_ID	READ: Device ID.
MPU9250_MAG_REG_INFORMATION	READ: Information.
MPU9250_MAG_REG_STATUS_1	READ: Data status.
MPU9250_MAG_REG_X_DATA_LSB	READ: X-axis data (LSB)
MPU9250_MAG_REG_X_DATA_MSB	READ: X-axis data (MSB)
MPU9250_MAG_REG_Y_DATA_LSB	READ: Y-axis data (LSB)
MPU9250_MAG_REG_Y_DATA_MSB	READ: Y-axis data (MSB)
MPU9250_MAG_REG_Z_DATA_LSB	READ: Z-axis data (LSB)
MPU9250_MAG_REG_Z_DATA_MSB	READ: Z-axis data (MLSB)
MPU9250_MAG_REG_STATUS_2	READ: Data Status.
MPU9250_MAG_REG_CONTROL	READ/WRITE: Mode Setting.
MPU9250_MAG_REG_SELF_TEST	READ/WRITE:
MPU9250_MAG_REG_I2C_DISABLE	READ/WRITE:
MPU9250_MAG_REG_X_SENSITIVITY	READ:
MPU9250_MAG_REG_Y_SENSITIVITY	READ:
MPU9250_MAG_REG_Z_SENSITIVITY	READ:

5.1.3.5 MPU9250_TEMP_REGISTER_t

enum MPU9250_TEMP_REGISTER_t

Enumerator

MPU9250 TEMP OUT MSB	Γ
MPU9250 TEMP OUT LSB	

5.1.4 Variable Documentation

5.1.4.1 MPU9250_ACCEL_REGISTER_t

enum { ... } MPU9250_ACCEL_REGISTER_t

5.2 NSL ThinSat.h File Reference

Function prototypes, includes, and definitions for NSL to TSLPB Arduino interface.

Macros

• #define NSL_PACKET_SIZE 38

Total bytes in the TSL Payload Packet.

• #define NSL_PACKET_HEADER_LENGTH 3

Total Bytes.

#define NSL_PACKET_HEADER {0x50, 0x50, 0x50}

The 3 byte preabmpe to NSL Payload Packets.

• #define NSL BAUD RATE 38400

From ETSat_Payload_ICD_v5.9.pdf page 10.

- #define NSL SERIAL ACK {0xAA, 0x05, 0x00}
- #define NSL_SERIAL_NAK {0xAA, 0x05, 0xFF}
- #define NSL_SERIAL_READY LOW

The NSL Mothership is able to receive a payload data packet.

• #define NSL_SERIAL_BUSY HIGH

The NSL Mothership is unable to receive a payload data packet.

5.2.1 Detailed Description

Function prototypes, includes, and definitions for NSL to TSLPB Arduino interface.

Author

Nicholas Counts

Date

06/12/18 This header is used by TSLPB.h and TSLPB.cpp to define the interface to the NSL Mothership.

5.2.2 Macro Definition Documentation

5.2.2.1 NSL_BAUD_RATE

#define NSL_BAUD_RATE 38400

From ETSat_Payload_ICD_v5.9.pdf page 10.

5.2.2.2 NSL_PACKET_HEADER

```
#define NSL_PACKET_HEADER {0x50, 0x50, 0x50}
```

The 3 byte preabmpe to NSL Payload Packets.

5.2.2.3 NSL_PACKET_HEADER_LENGTH

```
#define NSL_PACKET_HEADER_LENGTH 3
```

Total Bytes.

5.2.2.4 NSL_PACKET_SIZE

```
#define NSL_PACKET_SIZE 38
```

Total bytes in the TSL Payload Packet.

5.2.2.5 NSL_SERIAL_ACK

```
#define NSL_SERIAL_ACK {0xAA, 0x05, 0x00}
```

5.2.2.6 NSL_SERIAL_BUSY

```
#define NSL_SERIAL_BUSY HIGH
```

The NSL Mothership is unable to receive a payload data packet.

5.2.2.7 NSL_SERIAL_NAK

```
#define NSL_SERIAL_NAK {0xAA, 0x05, 0xFF}
```

5.2.2.8 NSL_SERIAL_READY

```
#define NSL_SERIAL_READY LOW
```

The NSL Mothership is able to receive a payload data packet.

5.3 ThinSat_DataPacket.h File Reference

Defines the custom data structure used to store the user's payload data, and the union that is used to transmit the data to the NSL Mothership.

Classes

• struct UserDataStruct_t

A user-customizable structure to hold any data the user intends to send back to Earth.

· union ThinsatPacket_t

A union of the UserDataStruct_t payloadData and a byte array that is used to send the user's mission data to the NSL Mothership.

5.3.1 Detailed Description

Defines the custom data structure used to store the user's payload data, and the union that is used to transmit the data to the NSL Mothership.

Author

Nicholas Counts

Date

06/20/18

5.4 TSLPB.cpp File Reference

Implementation of TSLPB interface for Arduino.

```
#include "TSLPB.h"
```

5.4.1 Detailed Description

Implementation of TSLPB interface for Arduino.

Author

Nicholas Counts

Date

06/12/18

5.5 TSLPB.h File Reference

Function prototypes, includes, and definitions for TSLPB Arduino interface.

```
#include "WProgram.h"
#include "avr/sleep.h"
#include "Wire.h"
#include "NSL_ThinSat.h"
#include "ThinSat_DataPacket.h"
#include "MPU9250_REGS.h"
```

Classes

• class TSLPB

The controller class for the TSL Payload Board. Create an instance of this class to use its member functions for accessing the onboard analog and digital sensors. Methods for communicating with the NSL Mothership are also included.

Macros

```
    #define TSL_SERIAL_STATUS_PIN 4
        NSL Serial Busy Line monitoring pin.

    #define TSL_ADC A7
        ADC reading the MUX_Output.
```

• #define TSL_MUX_A 7

Mux A - TSLPB pin number.

• #define TSL_MUX_B 8

Mux B - TSLPB pin number.

• #define TSL_MUX_C 9

Mux C - TSLPB pin number.

• #define TSL_MUX_RESPONSE_TIME 10

10 miliseconds to change

• #define TSL_SENSOR_READY_TIMEOUT 100

number of milliseconds to wait for an I2C device to become ready

• #define LMA_TEMP_REG_UNUSED_LSBS 5

TSLPB Digital Temperature Sensor (LMA75A) Macros.

• #define LMA_TEMP_REG_SIGN_BIT 9

The bit that contains indicates the sign. 0-based.

• #define LMA_TEMP_REG_DEGREES_PER_LSB 0.125

Enumerations

```
enum TSLPB_AnalogSensor_t {
 Solar = 0b000, IR = 0b001, TempInt = 0b010, TempExt = 0b011,
 Current = 0b100, Voltage = 0b101 }
     TSLPB Analog Sensor Selection Enum.
enum TSLPB_I2CAddress_t {
 DT1 ADDRESS = 0x4A, DT2 ADDRESS = 0x4C, DT3 ADDRESS = 0x4D, DT4 ADDRESS = 0x48,
 DT5 ADDRESS = 0x49, DT6 ADDRESS = 0x4B, IMU ADDRESS = 0x69, MAG ADDRESS = 0x0C }
     TSLPB Digital Sensor Address enum. Used by TSLPB private methods to communicate with the digital sensors over
    12C.
enum TSLPB_DigitalSensor_t {
 DT1, DT2, DT3, DT4,
 DT5, DT6, Accelerometer x, Accelerometer y,
 Accelerometer_z, Gyroscope_x, Gyroscope_y, Gyroscope_z,
 Magnetometer_x, Magnetometer_y, Magnetometer_z, IMU_Internal_Temp }
     TSLPB Digital Sensor selection Enum. Used as arguments for TSLPB::readDigitalSensor() and TSLPB::readDigitalSensorRaw()
enum LM75A_REG {
 LM75A_TEMPERATURE = 0x0, LM75A_CONFIGURATION = 0x1, LM75A_T_HYST = 0x2, LM75A_T_OS
 = 0x3,
 LM75A_PRODUCT_ID = 0x7 }
     TSLPB Digital Temperature Sensor (LMA75A) Register Selection Enum.
```

5.5.1 Detailed Description

Function prototypes, includes, and definitions for TSLPB Arduino interface.

Author

Nicholas Counts

Date

06/12/18

5.5.2 Macro Definition Documentation

5.5.2.1 LMA_TEMP_REG_DEGREES_PER_LSB

```
#define LMA_TEMP_REG_DEGREES_PER_LSB 0.125
```

Temperature resolution in °C per LSb.

```
5.5.2.2 LMA_TEMP_REG_SIGN_BIT
```

```
#define LMA_TEMP_REG_SIGN_BIT 9
```

The bit that contains indicates the sign. 0-based.

```
5.5.2.3 LMA_TEMP_REG_UNUSED_LSBS
```

```
#define LMA_TEMP_REG_UNUSED_LSBS 5
```

TSLPB Digital Temperature Sensor (LMA75A) Macros.

The number of bits to be discarded (from LSb)

5.5.2.4 TSL_ADC

#define TSL_ADC A7

ADC reading the MUX_Output.

5.5.2.5 TSL_MUX_A

#define TSL_MUX_A 7

Mux A - TSLPB pin number.

5.5.2.6 TSL_MUX_B

#define TSL_MUX_B 8

Mux B - TSLPB pin number.

5.5.2.7 TSL_MUX_C

#define TSL_MUX_C 9

Mux C - TSLPB pin number.

5.5 TSLPB.h File Reference 29

5.5.2.8 TSL_MUX_RESPONSE_TIME

#define TSL_MUX_RESPONSE_TIME 10

10 miliseconds to change

5.5.2.9 TSL_SENSOR_READY_TIMEOUT

```
#define TSL_SENSOR_READY_TIMEOUT 100
```

number of milliseconds to wait for an I2C device to become ready

5.5.2.10 TSL_SERIAL_STATUS_PIN

#define TSL_SERIAL_STATUS_PIN 4

NSL Serial Busy Line monitoring pin.

5.5.3 Enumeration Type Documentation

5.5.3.1 LM75A_REG

enum LM75A_REG

TSLPB Digital Temperature Sensor (LMA75A) Register Selection Enum.

Enumerator

LM75A_TEMPERATURE	0x00 Read only
LM75A_CONFIGURATION	0x01 Read/Write
LM75A_T_HYST	0x02 Read/Write
LM75A_T_OS	0x03 Read/Write
LM75A_PRODUCT_ID	0x07 Read only

5.5.3.2 TSLPB_AnalogSensor_t

enum TSLPB_AnalogSensor_t

TSLPB Analog Sensor Selection Enum.

Enumerator

Solar	0b000 (Solar Sensor)
IR	0b001 (IR)
TempInt	0b010 (Temp Int)
TempExt	0b011 (Temp Ext)
Current	0b100 (Current)
Voltage	0b101 (Vcc)

5.5.3.3 TSLPB_DigitalSensor_t

enum TSLPB_DigitalSensor_t

TSLPB Digital Sensor selection Enum. Used as arguments for TSLPB::readDigitalSensor() and TSLPB::readDigitalSensorRaw()

Enumerator

DT1	Select LM75A DT1.
DT2	Select LM75A DT2.
DT3	Select LM75A DT3.
DT4	Select LM75A DT4.
DT5	Select LM75A DT5.
DT6	Select LM75A DT6.
Accelerometer_x	Select MPU-9250 Accelerometer x-axis.
Accelerometer_y	Select MPU-9250 Accelerometer y-axis.
Accelerometer_z	Select MPU-9250 Accelerometer z-axis.
Gyroscope_x	Select MPU-9250 Gyroscope x-axis.
Gyroscope_y	Select MPU-9250 Gyroscope y-axis.
Gyroscope_z	Select MPU-9250 Gyroscope z-axis.
Magnetometer_x	Select MPU-9250 Magnetometer x-axis.
Magnetometer_y	Select MPU-9250 Magnetometer y-axis.
Magnetometer_z	Select MPU-9250 Magnetometer z-axis.
IMU_Internal_Temp	Select MPU-9250 Internal Temperature.

5.5.3.4 TSLPB_I2CAddress_t

enum TSLPB_I2CAddress_t

TSLPB Digital Sensor Address enum. Used by TSLPB private methods to communicate with the digital sensors over I2C.

Note

May be used by client code to access any of the I2C devices on the TSLPB. (with caution!)

Enumerator

DT1_ADDRESS	LM75A.
DT2_ADDRESS	LM75A.
DT3_ADDRESS	LM75A.
DT4_ADDRESS	LM75A.
DT5_ADDRESS	LM75A.
DT6_ADDRESS	LM75A.
IMU_ADDRESS	MPU-9250.
MAG_ADDRESS	MAGNETOMETER I2C Address (slave on the MPU-9250)

Index

ACC_FULL_SCALE_16_G	MPU9250_REGS.h, 20
MPU9250_REGS.h, 18	MPU9250_ACCEL_REGISTER_t
ACC_FULL_SCALE_2_G	MPU9250_REGS.h, 22
MPU9250_REGS.h, 19	MPU9250_GYRO_REGISTER_t
ACC_FULL_SCALE_4_G	MPU9250_REGS.h, 21
MPU9250_REGS.h, 19	MPU9250_MAG_CONTROL_t
ACC_FULL_SCALE_8_G	MPU9250_REGS.h, 21
MPU9250_REGS.h, 19	MPU9250_MAG_REGISTER_t
	MPU9250_REGS.h, 22
begin	MPU9250_REGS.h, 17
TSLPB, 8	ACC_FULL_SCALE_16_G, 18
bmePres	ACC_FULL_SCALE_2_G, 19
UserDataStruct_t, 13	ACC_FULL_SCALE_4_G, 19
bmeTemp	ACC_FULL_SCALE_8_G, 19
UserDataStruct_t, 13	GYRO_FULL_SCALE_1000_DPS, 19
bnoCal	GYRO_FULL_SCALE_2000_DPS, 19
UserDataStruct_t, 13	GYRO_FULL_SCALE_250_DPS, 19
bnomagx	GYRO_FULL_SCALE_500_DPS, 20
UserDataStruct_t, 13	MAG_MAX_BYTE_VALUE, 20
bnomagy	MAG_MAX_VALUE_FLOAT, 20
UserDataStruct_t, 13	MPU9250_ACCEL_REGISTER_t, 22
bnomagz	MPU9250_GYRO_REGISTER_t, 21
UserDataStruct_t, 13	MPU9250_MAG_CONTROL_t, 21
CVDO FILL COME 1000 DDC	MPU9250_MAG_REGISTER_t, 22
GYRO_FULL_SCALE_1000_DPS	MPU9250_TEMP_REGISTER_t, 22
MPU9250_REGS.h, 19 GYRO FULL SCALE 2000 DPS	MPU9250_TEMP_REGISTER_t
MPU9250 REGS.h, 19	MPU9250_REGS.h, 22
GYRO FULL SCALE 250 DPS	
MPU9250 REGS.h, 19	NSL_BAUD_RATE
GYRO FULL SCALE 500 DPS	NSL_ThinSat.h, 23
MPU9250 REGS.h, 20	NSL_PACKET_HEADER_LENGTH
MF 09230_REG3.11, 20	NSL_ThinSat.h, 24
header	NSL_PACKET_HEADER
UserDataStruct_t, 14	NSL_ThinSat.h, 23
	NSL_PACKET_SIZE
isClearToSend	NSL_ThinSat.h, 24
TSLPB, 9	NSL_SERIAL_ACK
	NSL_ThinSat.h, 24
LM75A_REG	NSL_SERIAL_BUSY
TSLPB.h, 29	NSL_ThinSat.h, 24
LMA_TEMP_REG_DEGREES_PER_LSB	NSL_SERIAL_NAK
TSLPB.h, 27	NSL_ThinSat.h, 24
LMA_TEMP_REG_SIGN_BIT	NSL_SERIAL_READY
TSLPB.h, 27	NSL_ThinSat.h, 24 NSL ThinSat.h, 23
LMA_TEMP_REG_UNUSED_LSBS	<u> </u>
TSLPB.h, 28	NSL_BAUD_RATE, 23 NSL PACKET HEADER LENGTH, 2
MAG_MAX_BYTE_VALUE	
MPU9250 REGS.h, 20	NSL_PACKET_HEADER, 23 NSL_PACKET_SIZE, 24
MAG_MAX_VALUE_FLOAT	NSL SERIAL ACK, 24
IVIAU_IVIAA_VALUE_FLUAT	NOL_OLDIAL_AUN, 24

34 INDEX

NSL_SERIAL_BUSY, 24 NSL_SERIAL_NAK, 24 NSL_SERIAL_READY, 24 NSLPacket ThinsatPacket_t, 7	TSLPB_AnalogSensor_t, 29 TSLPB_DigitalSensor_t, 30 TSLPB_I2CAddress_t, 30 TSLPB_AnalogSensor_t TSLPB.h, 29 TSLPB_DigitalSensor_t
payloadData	TSLPB.h, 30
ThinsatPacket_t, 7	TSLPB_I2CAddress_t
pushDataToNSL	TSLPB.h, 30
TSLPB, 9	TSLPB, 8
	begin, 8
quatw	isClearToSend, 9
UserDataStruct_t, 14	pushDataToNSL, 9
quatx	read8bitRegister, 10
UserDataStruct_t, 14	readAnalogSensor, 10
quaty	readDigitalSensor, 10
UserDataStruct_t, 14	readDigitalSensorRaw, 11
quatz	sleepUntilClearToSend, 11
UserDataStruct_t, 14	ThinSat_DataPacket.h, 25
read8bitRegister	ThinsatPacket_t, 7
TSLPB, 10	NSLPacket, 7
readAnalogSensor	payloadData, 7 tslCurrent
TSLPB, 10	
readDigitalSensor	UserDataStruct_t, 14 tslMagXraw
TSLPB, 10	UserDataStruct_t, 15
readDigitalSensorRaw	tslMagYraw
TSLPB, 11	UserDataStruct_t, 15
	tslMagZraw
sleepUntilClearToSend	UserDataStruct_t, 15
TSLPB, 11	tslTempExt
	UserDataStruct_t, 15
TSL_ADC	tslVolts
TSLPB.h, 28	UserDataStruct_t, 15
TSL_MUX_RESPONSE_TIME	
TSLPB.h, 28	unused5
TSL_MUX_A	UserDataStruct_t, 15
TSLPB.h, 28 TSL MUX B	UserDataStruct_t, 11
TSLPB.h, 28	bmePres, 13
TSL MUX C	bmeTemp, 13
TSLPB.h, 28	bnoCal, 13
TSL_SENSOR_READY_TIMEOUT	bnomagx, 13
TSLPB.h, 29	bnomagy, 13 bnomagz, 13
TSL_SERIAL_STATUS_PIN	header, 14
TSLPB.h, 29	quatw, 14
TSLPB.cpp, 25	quatx, 14
TSLPB.h, 26	quaty, 14
LM75A_REG, 29	quatz, 14
LMA_TEMP_REG_DEGREES_PER_LSB, 27	tslCurrent, 14
LMA_TEMP_REG_SIGN_BIT, 27	tslMagXraw, 15
LMA_TEMP_REG_UNUSED_LSBS, 28	tslMagYraw, 15
TSL_ADC, 28	tslMagZraw, 15
TSL_MUX_RESPONSE_TIME, 28	tslTempExt, 15
TSL_MUX_A, 28	tslVolts, 15
TSL_MUX_B, 28	unused5, 15
TSL_MUX_C, 28	
TSL_SENSOR_READY_TIMEOUT, 29	
TSL_SERIAL_STATUS_PIN, 29	