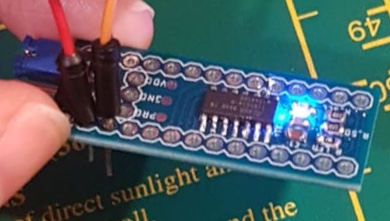
|  |
| --- |
| UKMARSBOT I2C Sensor Controller Interface Specification |



Features

* Connects directly between UKMARSBOT and existing UKMARS sensor boards to offer an I2C interface for sensors / LED indicators
* Supports UKMARS sensor boards that provide up to:
  + 5 x Analogue Sensor Inputs
  + 1 x Common Transmitter Output
  + 2 x Indicator LEDs
* Collects sensor data periodically, with configurable:
  + Ambient light removal
  + Sensor transmitter pulse length
  + ADC Resolution (8-bit / 10-bit)
  + Sensor selection – disable unused sensors for increased speed
* Threshold Comparison:
  + Compares each sensor value against a configurable threshold value to return Sensor State
  + Optional polarity inversion
* Configurable Interrupt Source (may be polled):
  + Rising and/or Falling Sensor state change
    - Individually Settable for each Sensor
  + Sensor board specific interrupt
* Configurable Indicator LED control:
  + Master Control (over I2C)
  + Sample Frequency output
  + Board specific indication
* Selectable Operating Voltage
  + 3.3V or 5V
* Sensor board specific features and Interrupts
  + Selectable over I2C

Functional Block Diagram of Application

Below is a diagram showing the application in which the I2C Sensor Controller is intended to be used.

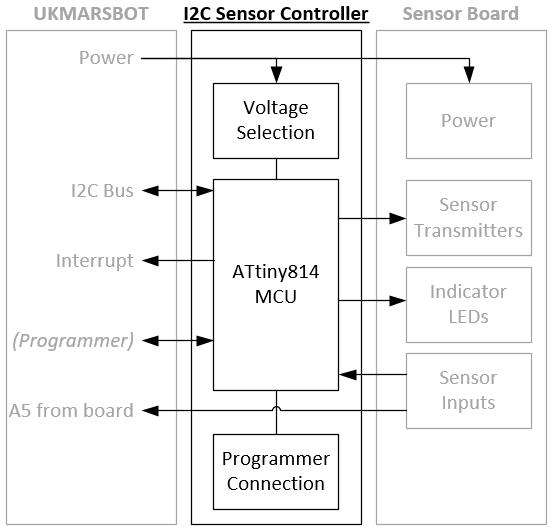


Table of Contents

[1.0 ELECTRICAL CHARACTERISTICS 3](#_Toc62054049)

[2.0 PIN DESCRIPTIONS 3](#_Toc62054050)

[3.0 DEVICE OVERVIEW 3](#_Toc62054051)

[4.0 POWER-ON SEQUENCE 4](#_Toc62054052)

[5.0 SERIAL INTERFACE (I2C) 4](#_Toc62054053)

[6.0 PERFORMANCE 5](#_Toc62054054)

[7.0 REGISTER SUMMARY 6](#_Toc62054055)

[8.0 REGISTER DESCRIPTIONS 7](#_Toc62054056)

[9.0 Line Sensor Registers 19](#_Toc62054057)

[10.0 Firmware Revision History 24](#_Toc62054058)

[11.0 Nomenclature 25](#_Toc62054059)

[12.0 APPLICATION NOTES: 26](#_Toc62054060)

[13.0 Adding a new register: 27](#_Toc62054061)

# PIN DESCRIPTIONS

The descriptions of the pins as used by the MCU are listed in Table 1.

Table 1 - PINOUT DESCRIPTION

|  |  |  |
| --- | --- | --- |
| **UKMARSBOT Pin Name** | **UKMARSBOT Pin Mode** | **Function** |
| 5V | Power | Provides 5V to equivalent Sensor Board pin |
| GND | Power | Provides Ground to equivalent Sensor Board pin |
| A5 | I2C SCL | I2C Serial Clock (with optional pullup resistor attached) |
| A4 | I2C SDA | I2C Serial Data (with optional pullup resistor attached) |
| A3 | UPDI | Used only to flash program to the ISC when using Arduino as UPDI programmer |
| A2 | Analog Input | Directly connected to A5 pin on Sensor Board |
| A1 | Digital Input | Interrupt output - configurable |
| *A0* | *NC* | *May use elsewhere on UKMARSBOT* |
| *D11* | *NC* | *May use elsewhere on UKMARSBOT* |
| *D12* | *NC* | *May use elsewhere on UKMARSBOT* |
| *D6* | *NC* | *May use elsewhere on UKMARSBOT* |
| GND | Power | Ground |
| 5V | Power\* | Provides 5V to equivalent Sensor Board pin |
| 3V3 | Power\* | Provides 3.3V to equivalent Sensor Board pin |

\* The Voltage Selection header selects between these signals to supply power to the ISC, this defines the logic levels and must match that of the MCU.

# DEVICE OVERVIEW

The Intelligent Sensor Controller (ISC) handles the reading and interpretation of UKMARS sensor data and provides sensor values and states to the UKMARSBOT Mainboard over an I2C bus. The device consists of an ATTiny814 controlling the UKMARS Sensor board in a manner configured by the UKMARSBOT MCU. Registers are used to configure and obtain data for each sensor.

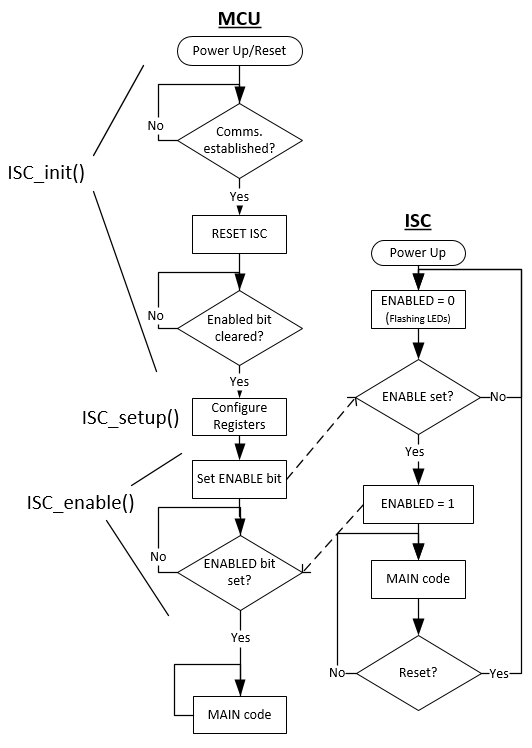
The sensor is dynamically compared to the associated threshold register (SENS#THRSH) with the Polarity Inversion register determining the state of the pin. All registers can be read by the system master. The interrupt pin indicates that an event has occurred, this can be configured to activate under the following conditions:

1. When any of the (masked) sensor states differ from its previous state. This is used to indicate to the system master that an input state has changed.

# POWER-ON SEQUENCE

It is recommended that the ISC is connected to the UKMARSBOT and Sensor board prior to the supply of power. Once power is supplied, the ISC must be configured. Furthermore, any subsequent loss of power requires the configuration to be set again.

The following flow diagrams illustrate the relationship between MCU and ISC on Power-up or MCU reset.



1. When first connected to power, the ENABLED bit is LOW, indicating that registers must be configured by the MCU.
2. Once all registers have been configured, the MCU sets the ENABLE bit in the ISC.
3. The MCU waits for the ENABLED bit to be set, confirming that ISC setup is complete.
4. MCU enters the main code.

It is recommended that the MCU always issue a RESET command after start-up in the event that the power has remained on, but the MCU has been reset. This forces the ISC back into a power-up state.

# SERIAL INTERFACE (I2C)

The ISC contains individual 8-bit registers (some of which form pairs) that can be addressed through the Serial Interface at address 0x50 (7-bit). An overview of these may be found in Section 7.0 REGISTER SUMMARY. With further details located in Section 8.0 REGISTER DESCRIPTION.

The ISC automatically increments the address pointer after sending each byte during the data transfer. The address pointer automatically rolls over to address 00h after accessing the last register. This may be utilised when reading consecutive registers such as Sensor Values. Simply write the first register address, then read, with repeated starts to retrieve consecutive bytes.

The ISC has been tested at 400 kHz clock speed with no adverse effects.

Refer to **Section 26. TWI – Two Wire Interface** of the ATTiny814 Manual for detailed requirements:

<http://ww1.microchip.com/downloads/en/DeviceDoc/40001912A.pdf>

# PERFORMANCE

The following tests have been conducted to provide an indication of the performance that is to be expected from the ISC. All were performed with an Arduino Nano V3.0 Clone fitted onto a standard UKMARS Chassis with the ISC located between the UKMARSBOT and Basic Line Sensor board, operating at 5V.

The following remained constant for all experiments:

* All registers contain their default values other than what is detailed below.
* LED control was set to FREQOUT and a PicoScope2204 was connected to measure the loop frequency

## Read 1 Sensor RAW value

Sensor 1 enabled.

## Read 1 Sensor with Tx LED at ≥20µs

## Read 6 Sensor’s RAW values

## Read 6 Sensors with Tx LED at ≥20µs

1 Sensor active with threshold comparison. 664us

Value / Counts per second = time (approx. 0.52ms for all 6 sensors).

# ISC REGISTER SUMMARY

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **OFFSET** | **Name** | **Bit Pos** |  |  |  |  |  |  |  |  |
| 0 (0x00) | FIRMVER | 7:0 | FIRMWARE\_VERSION | | | | | | | |
| 1 (0x01) | CONTROL | 7:0 | ENABLED |  |  | BOARD TYPE | | | RESET | ENABLE |
| 2 (0x02) | INTERRUPT CTRL | 7:0 | BRDINTEN |  |  | SENS4 INTEN | SENS3 INTEN | SENS2 INTEN | SENS1 INTEN | SENS0 INTEN |
| 3 (0x03) | INTERRUPT FLAGS | 7:0 | BRDINT |  |  | SENS4 INT | SENS3 INT | SENS2 INT | SENS1 INT | SENS0 INT |
| 4 (0x04) | SENSSTATE | 7:0 |  |  |  | SENS4 STATE | SENS3 STATE | SENS2 STATE | SENS1 STATE | SENS0 STATE |
| 5 (0x05) | LEDCTRL | 7:0 | LLED | RLED |  |  |  | INDICATION\_MODE | | |
| 6 (0x06) | PULSEDUR | 7:0 | PULSE\_DURATION (µs) | | | | | | | |
| 7 (0x07) | SENS0SETUP | 7:0 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| 8 (0x08) | SENS1SETUP | 7:0 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| 9 (0x09) | SENS2SETUP | 7:0 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| 10 (0x0A) | SENS3SETUP | 7:0 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| 11 (0x0B) | SENS4SETUP | 7:0 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| 12 (0x0C) | Reserved | 7:0 |  |  |  |  |  |  |  |  |
| 13 (0x0D) | SENS0VAL | 15:8 | MSByte of SENS0VAL | | | | | | | |
| 14 (0x0E) | 7:0 | LSByte of SENS0VAL | | | | | | | |
| 15 (0x0F) | SENS1VAL | 15:8 | MSByte of SENS1VAL | | | | | | | |
| 16 (0x10) | 7:0 | LSByte of SENS1VAL | | | | | | | |
| 17 (0x011) | SENS2VAL | 15:8 | MSByte of SENS2VAL | | | | | | | |
| 18 (0x12) | 7:0 | LSByte of SENS2VAL | | | | | | | |
| 19 (0x13) | SENS3VAL | 15:8 | MSByte of SENS3VAL | | | | | | | |
| 20 (0x14) | 7:0 | LSByte of SENS3VAL | | | | | | | |
| 21 (0x15) | SENS4VAL | 15:8 | MSByte of SENS4VAL | | | | | | | |
| 22 (0x16) | 7:0 | LSByte of SENS4VAL | | | | | | | |
| 23 (0x17) | Reserved | 15:8 |  | | | | | | | |
| 24 (0x18) | 7:0 |  | | | | | | | |
| 25 (0x19) | SENS0THRSH | 15:8 | MSByte of SENS0THRSH | | | | | | | |
| 26 (0x1A) | 7:0 | LSByte of SENS0THRSH | | | | | | | |
| 27 (0x1B) | SENS1THRSH | 15:8 | MSByte of SENS1THRSH | | | | | | | |
| 28 (0x1C) | 7:0 | LSByte of SENS1THRSH | | | | | | | |
| 29 (0x1D) | SENS2THRSH | 15:8 | MSByte of SENS2THRSH | | | | | | | |
| 30 (0x1E) | 7:0 | LSByte of SENS2THRSH | | | | | | | |
| 31 (0x1F) | SENS3THRSH | 15:8 | MSByte of SENS3THRSH | | | | | | | |
| 32 (0x20) | 7:0 | LSByte of SENS3THRSH | | | | | | | |
| 33 (0x21) | SENS4THRSH | 15:8 | MSByte of SENS4THRSH | | | | | | | |
| 34 (0x22) | 7:0 | LSByte of SENS4THRSH | | | | | | | |
| 35 (0x23) | Reserved | 15:8 |  | | | | | | | |
| 36 (0x24) | 7:0 |  | | | | | | | |
| 37 (0x25) | SCANTIME | 15:8 | MSByte of SCANTIME | | | | | | | |
| 38 (0x26) | 7:0 | LSByte of SCANTIME | | | | | | | |
| 39 (0x27) | Board Specific Registers – More details in specific board section. | | | | | | | | | |
| … |

# ISC REGISTER DESCRIPTIONS

## Firmware Version

* Name: FIRMVER
* Offset: 0x00
* Reset: 0x##

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x00 | FIRMWARE\_VERSION[7:0] | | | | | | | |
| Access | R | | | | | | | |
| Reset | ## | | | | | | | |

Bits 7:0 – FIRMWARE\_VERSION[7:0]

Firmware version installed on the ISC. See Section *8.0 Firmware Revision History* for more detailed information regarding the current firmware version.

## Control

* Name: CONTROL
* Offset: 0x01
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x01 | ENABLED |  |  | BOARD\_TYPE[2:0] | | | RESET | ENABLE |
| Access | R |  |  | R/W | | | W | W |
| Reset | 0 |  |  | 0 | | | 0 | 0 |

Bit 7 – ENABLED

Status of board operation:

Set HIGH when the ENABLE is HIGH

Set LOW when RESET is HIGH

Bits 4:2 – BOARD\_TYPE[2:0]

Defines the board that is connected to the ISC:

|  |  |
| --- | --- |
| **Value** | **Board** |
| 0 (0x00) | Undefined |
| 1 (0x01) | UKMARS Basic Line Sensor |
| 2 (0x02) | UKMARS Basic Wall Sensor |
| 3 (0x03) | S. Pithouse Line Sensor |

Bit 1 – RESET

Set HIGH to reset the sensor board. Clears to a 0 immediately. This sets ENABLED LOW.

Bit 0 – ENABLE

Set HIGH when to start the main sequence. Clears to a 0 immediately. This sets ENABLED HIGH.

## Interrupt Control

* Name: INTERRUPTCTRL
* Offset: 0x02
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x02 | BRD INTEN |  |  | SENS4 INTEN | SENS3 INTEN | SENS2 INTEN | SENS1 INTEN | SENS0 INTEN |
| Access | R/W |  |  | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 |  |  | 0 | 0 | 0 | 0 | 0 |

Bit 7 – BRDINTEN

Set HIGH to enable interrupts from *BRDINTFLAG* register.

Bit 4-0 – SENS4INTEN – SENS0INTEN

Set HIGH to enable an interrupt on the corresponding sensor changing state.

NOTE that the interrupt bit must be set in *SENS#SETUP* to enable the specific sensor to trigger an interrupt.

## Interrupt Flags

* Name: INTERRUPTFLAGS
* Offset: 0x03
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x03 | BRD INT |  |  | SENS4 INT | SENS3 INT | SENS2 INT | SENS1 INT | SENS0 INT |
| Access | R |  |  | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 |  |  | 0 | 0 | 0 | 0 | 0 |

Bit 7 – BOARD

HIGH when a bit is set in *BRDINTFLAGS.*

LOW when all bits in *BRDINTFLAGS* are LOW.

See *BRDINTFLAGS* for details on how to clear interrupt bits.

Bit 4-0 – SENS#STATE

Set HIGH internally by the ISC if the following conditions are met:

|  |  |  |
| --- | --- | --- |
| SENS#INTEN set HIGH in INTERRUPT\_CTRL | | |
| R\_INT\_ENB set HIGH in SENS#SETUP | OR | F\_INT\_ENB set HIGH in SENS#SETUP |
| SENS#VAL has risen above SENS#THRSH | SENS#VAL has fallen below SENS#THRSH |

Must be reset externally. To reset, write a 1 to the corresponding bit location.

## Sensor State

* Name: SENSSTATE
* Offset: 0x04
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x04 |  |  |  | SENS4STATE | SENS3STATE | SENS2STATE | SENS1STATE | SENS0STATE |
| Access |  |  |  | R | R | R | R | R |
| Reset |  |  |  | 0 | 0 | 0 | 0 | 0 |

Bit 4-0 – SENS#STATE

Result of most recent comparison between *SENS#VAL* and *SENS#THRSH* registers.

HIGH if SENS#VAL > SENS#THRSH.

If the corresponding interrupt flag is on, the state remains frozen until the interrupt is cleared. This allows the state to be interrogated to determine the direction of the interrupt.

## Indicator LED Control

* Name: LEDCTRL
* Offset: 0x05
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x05 | LLED | RLED |  |  |  | INDICATION\_MODE[2:0] | | |
| Access | R/W | R/W |  |  |  | R/W | | |
| Reset | 0 | 0 |  |  |  | 0x00 | | |

Bit 7 - LLED

Returns current state of the Left Indicator LED. Where 0: OFF and 1: ON.

The Left Indicator LED state may be set by writing a 0 or 1 to this bit when *INDICATION\_MODE* is set to MASTER CONTROL.

Bit 6 - RLED

Returns current state of the Right Indicator LED. Where 0: OFF and 1: ON.

The Right Indicator LED state may be set by writing a 0 or 1 to this bit when *INDICATION\_MODE* is set to MASTER CONTROL.

Bits 2:0 – INDICATION\_MODE[2:0]

|  |  |  |
| --- | --- | --- |
| **Value** | **Description** | |
| 0 | MASTER CONTROL | Indicator LEDs are controlled by corresponding bits *LLED* and *RLED* |
| 1 | FREQOUT | Toggles LED state each sample cycle – this can be used to determine the sample rate |
| 2 | OFF | Force both indicator LEDs OFF |
| 3 | BRDCONTROL | Indicator LEDs reflect state specific to the attached Sensor board. See BRDCONFIG register for details |
| 4 | FASTBLINK | Indicator LEDs flash quickly until INDICATION MODE value is changed |
| 5 | SLOWBLINK | Indicator LEDs flash slowly until INDICATION MODE value is changed |

## Pulse Duration

* Name: PULSEDUR
* Offset: 0x06
* Reset: 0x0A

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x06 | PULSE\_DURATION[7:0] | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0x0A | | | | | | | |

Bits 7:0 – PULSE\_DURATION[7:0]

Minimum time (in µs) that emitter is on before an ADC is performed. This is only used when TXEN bit is TRUE in corresponding SENS#SETUP.

## Sensor Setup

* Name: SENS#SETUP
* Offset: 0x07 –> 0x0B
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x07 | R\_INTENB | F\_INTENB | POLARITY |  |  | TX\_EN | RESOL | ENB |
| Access | R/W | R/W | R/W |  |  | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 |  |  | 0 | 0 | 0 |

Bit 7 – R\_INTENB

Enables ability for Sensor to trigger an interrupt on a rising edge change – note that sensor state interrupt must be enabled in *INTCTRL* Register.

Bit 6 - F\_INTENB

Enables ability for sensor to trigger an interrupt on a falling edge change – note that sensor state interrupt must be enabled in *INTCTRL* Register.

Bit 5 - POLARITY

Not used when BOARD TYPE is defined. POLARITY is used to inform the ISC whether a low Analog Read value represents white or black.

0: Low analog read value = Black??

1: Low analog read value = White??

Bit 2 – TX\_EN

0: Disables the use of the Emitter LED when sampling the sensor. The selected sensor is sampled with the LED turned off and the result is returned in *SENS#VAL* Register.

1: Enables the use of the Sensor Transmitter. The sensors will be sampled, then the Emitter turned on for a duration (in units of µs) set by *PULSEDUR* register before being sampled again. The difference between these is returned in *SENS#VAL* Register.

Bit 1 – RESOLUTION

0: *SENS#VAL* Register pair contains 10-bit result from sensor

1: *SENS#VAL* Register pair contains 8-bit result from sensor – hence only necessary to read the LSB register. Note that the ADC sample resolution remains 10-bit, it is converted to 8-bit when stored in the register.

Bit 0 - ENB

When TRUE, the sensor will be sampled each scan, set this to FALSE if sensor is not being used to reduce time between consecutive samples

## Sensor Value

* Name: SENS#VAL
* Offset: 0x0D -> 0x16
* Reset: 0x0000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x0D | MSByte of SENS0VAL | | | | | | | |
| 0x0E | LSByte of SENS0VAL | | | | | | | |
| 0x0F – 0x16 | MSByte/LSByte of SENS1-4VAL | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0x00 | | | | | | | |

Addresses 0x0D to 0x16 allow the most recent sensor value to be read for each sensor, starting with Sensor 0. Values are up to 10-bits long, hence are organised across two registers.

When the MSB register is read, the corresponding I2C register becomes ‘frozen’ so that the LSB register refers to the same value. This prevents the value getting updated between MSB and LSB reads.

After the LSB register has been read, the value becomes ‘unfrozen’, allowing it to update before the next read.

## Sensor Threshold

* Name: SENS#THRSH
* Offset: 0x19 -> 0x22
* Reset: 0x0000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x19 | MSByte of SENS0THRSH | | | | | | | |
| 0x1A | LSByte of SENS0THRSH | | | | | | | |
| 0x1B -0x22 | MSByte/LSByte of SENS1-4THRSH | | | | | | | |
| Access | R/W | | | | | | | |
| Reset | 0x0000 | | | | | | | |

Addresses 0x19 to 0x22 contain sensor threshold values associated with each sensor, starting with Sensor 0. Values may be up to 10-bits long, hence are organised across two registers.

Bits 15:0 – SENS0THRSH[15:0]

Threshold value which sensor value is compared against to determine the sensor state.

## Scan Time

* Name: SCANTIME
* Offset: 0x25 -> 0x26
* Reset: 0x0000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x25 | MSByte of SCANTIME[15:8] | | | | | | | |
| 0x26 | LSByte of SCANTIME[7:0] | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0x0000 | | | | | | | |

Bits 15:0 - SCANTIME[15:0]

Time (in µs) taken for the last complete cycle. May be used to monitor the Sensor refresh rate.

The value (0 µs to 65,535 µs) has a resolution of 1.2µs.

Note that in the event that a Timer overflow occurs, a value of 0xFFFF will be stored.

# LINE SENSOR REGISTER SUMMARY

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **OFFSET** | **Name** | **Bit Pos** |  |  |  |  |  |  |  |  |
| 39 (0x27) | BRDCONFIG | 7:0 | CALIB COMPL |  |  |  |  | BRD\_INDICATION\_MODE | | |
| 40 (0x28) | BRDINTCTRL | 7:0 |  |  |  |  |  |  | CROSS OVERINTEN | LOST LINEINTEN |
| 41 (0x29) | BRDINTFLAGS | 7:0 |  |  |  |  |  |  | CROSS OVERINT | LOST LINEINT |
| 42 (0x2A) | LINEERROR | 15:8 | MSByte of Line Error | | | | | | | |
| 43 (0x2B) | 7:0 | LSByte of Line Error | | | | | | | |

## Board Configuration

* Name: BRDCONFIG
* Offset: 0x27
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x27 | CALIB COMPL |  |  |  |  | BRD\_INDICATION\_MODE | | |
| Access | R |  |  |  |  | R/W | | |
| Reset | 0 |  |  |  |  | 0 | | |

Bit 7 – CALIBCOMPL

Gets set to TRUE once the initial line sensor calibration has completed

Bits 2:0 - BRD\_INDICATION\_MODE

When LEDCTRL INDICATION\_MODE value is set for BRDCONTROL, BRD\_INDICATION\_MODE value controls the status of the LEDs.

|  |  |  |
| --- | --- | --- |
| **Value** | **Description** | |
| 0 | SIDE | Indicator LEDs indicate the side of the line that the robot is on |
| 1 | MARKER Threshold | Indicator LEDs indicate the live state of the track marker sensors |
| 2 | L/R threshold | Left LED ON when left sensor above threshold, and right LED on when right sensor above threshold |

## Board Interrupt Control

* Name: BRDINTCTRL
* Offset: 0x28
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x28 |  |  |  |  |  |  | CROSS OVRINTEN | LOST LINEINTEN |
| Access |  |  |  |  |  |  | R/W | R/W |
| Reset | 0x00 | | | | | | | |

Bit 1 – CROSSOVERINTEN

Set HIGH to enable interrupts from *CROSSOVERINT* bit.

Bit 0 – LOSTLINEINTEN

Set HIGH to enable interrupts from *LOSTLINEINT* bit.

## Board Interrupt Flags

* Name: BRDINTFLAGS
* Offset: 0x29
* Reset: 0x00

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x29 |  |  |  |  |  |  | CROSS OVRINT | LOST LINEINT |
| Access |  |  |  |  |  |  | R/W | R/W |
| Reset |  |  |  |  |  |  | 0 | 0 |

Bit 2 – CROSSOVRINT

Set to TRUE when a crossover is detected – may be used to ignore marker sensors. Reset by writing a 1 to this location

Bit 0 – LOSTLINEINT

Set TRUE when the sensor cannot detect a line. Reset by writing a 1 to this location

## Line Error

* Name: Line Error
* Offset: 0x2A -> 0x2B
* Reset: 0x0000

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0x2A | MSByte of LINEERROR[15:8] | | | | | | | |
| 0x2B | LSByte of LINEERROR[7:0] | | | | | | | |
| Access | R | | | | | | | |
| Reset | 0x0000 | | | | | | | |

Bits 15:0 – LINEERROR

Distance from line. Note that a value of 512 represents no error (centred on line).

Reading 2-byte registers:

When the MSB register is read, the corresponding I2C register becomes ‘frozen’ so that the LSB register refers to the same value. This prevents the value getting updated between MSB and LSB reads.

After the LSB register has been read, the value becomes ‘unfrozen’, allowing it to update before the next read.

# NOMENCLATURE

**ISC** I2C Sensor Controller

**MCU** Microcontroller connected to the ‘Arduino Nano Style’ headers on the UKMARSBOT

# FIRMWARE REVISION HISTORY

Firmware revisions are detailed in Table 2.

Table 2 - Revision History

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Details** |
| 1 | 19/12/2020 | Release Version |
| 2 | 20/01/2021 | Added ENABLED Register – ENABLE is now to be a strobe |
| 3 | 22/01/2021 | Shuffled registers around |
| 4 | 25/01/2021 | Using RTC interrupt for LED flashing and Timer A for scanTime |
| 5 | 26/01/2021 | Reset interrupt output in receive function, set at end of loop |
| 6 | 01/02/2021 | Added blink modes in LEDCTRL and modified I2C sendData to update reg[] for all 2-byte values |

*Misc*

1. This layout is based on MCP23017 Datasheet from here: [https://ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdfhttps://ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdf](https://ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdfhttps:/ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdf)

From AVR manual:

An interrupt source is enabled or disabled by writing to the corresponding enable bit in the peripheral's Interrupt Control register (peripheral.INTCTRL).

An interrupt request is generated when the corresponding interrupt source is enabled and the Interrupt Flag is set. The interrupt request remains active until the Interrupt Flag is cleared. See the peripheral's INTFLAGS register for details on how to clear Interrupt Flags.

ALSO:

The TCA.PERL and TCA.PERH register pair represents the 16-bit value, TCA.PER. The low byte [7:0] (suffix L) is accessible at the original offset. The high byte [15:8] (suffix H) can be accessed at offset + 0x01. For more details on reading and writing 16-bit registers, refer to Accessing 16-bit Registers.

# ADDING A NEW REGISTER

The below points should be considered / implemented when adding a new register or modifying an existing one.

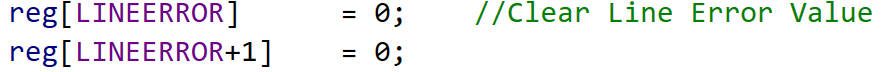
1. Define in addressMap.h



1. Update numRegisters in addressMap.h



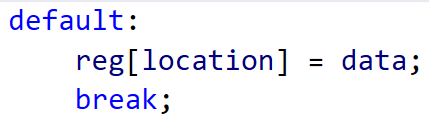
1. Update setRegDefaults() function



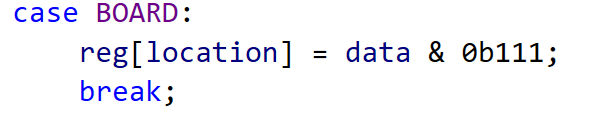
1. Update receiveData() function
   1. Readonly?



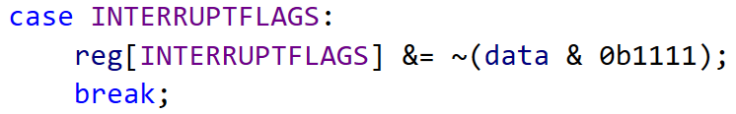
* 1. Write?
     1. If the whole register is to be replaced, ‘default’ case handles this.



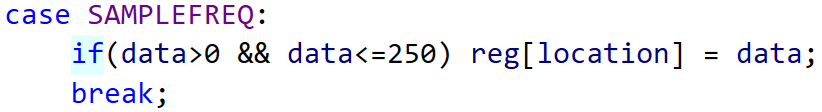
* + 1. If only some elements can be written to:



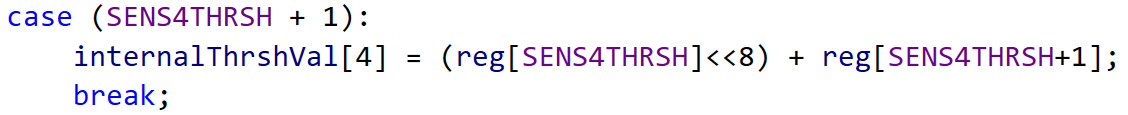
* + 1. If bits are to be cleared (ie. interrupt flags):



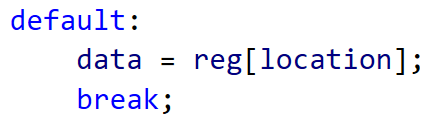
* + 1. If value is constrained:



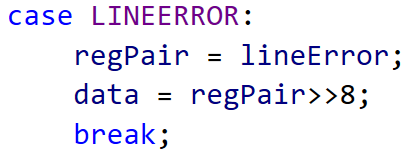
* 1. Second of two-byte pair is written, combine to create internal value.



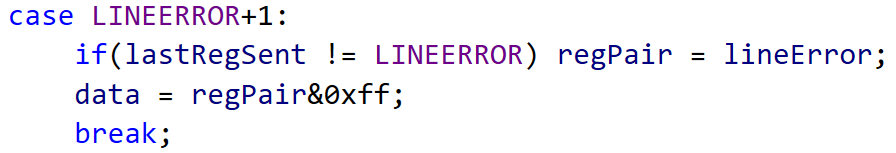
1. Update sendData() function
   1. If a single register is to be sent, ‘default’ case handles this:



* 1. If the first byte of a 2-byte pair is read:
     1. The corresponding internal value (MSB and LSB) is ‘frozen’ in regPair
     2. MSB is extracted to be sent



* 1. If the second byte of a 2-byte pair is read:
     1. If the last register sent was not the first byte of the pair, update the ‘frozen’ value
     2. LSB is extracted to be sent



# COMMISSIONING NOTES

Desired Functionality:

1. Read live sensor values
2. Ability to alter emitter pulse duration
3. Ability to read raw sensor value without emitter turned on
4. Set threshold of White/Black
5. Obtain sensor state (result of comparison on live value and threshold)
   1. ~~Option to invert polarity~~ (is this necessary with ability to choose from below?)
6. Interrupt on sensor state changing (RISING / FALLING / BOTH)
7. Add Register for control / configuration of onboard indicator LEDs:
   1. Indicating side of line
   2. Set over I2C registers
8. Be able to set sensor update frequency with register
9. ~~Be able to set I2C address from jumper on board – 3 bits is plenty (use DIP switch).~~
   1. ~~OR is it possible to write to non-volatile memory?~~ Is this necessary?

Software notes:

* Remove ability to invert sensor state in sensor setup – rising and falling interrupts remove the need for this I think
* Make code match manual, noting that registers have changed order, and some bits have been added/removed and moved around
* Remove loop frequency stuff – just loop as fast as possible
* Make timer time cycle and send over I2C – freezing LSB if required
* Make address = 0x50 (7-bit)
* Test TCA period to see what value scanTime becomes, tweak to prevent wraparound

Considerations for next version:

* Silkscreen could indicate pin numbers to reference to from documentation
* Consider using DI as interrupt –frees up analog Input pins
* Remove UPDI from UKMARSBOT interface? – is it easy to program UPDI from Arduino?
* Does Voltage selection header need to be a header, or could it be a solder bridge?
* Fix Silkscreen for V selection header
* Use readable size Silkscreen lettering
* Indicate LED polarity on silkscreen

# APPLICATION NOTES

Basic Line Sensor:

Corner / Radius markers: Rising Interrupt, enabled with Tx

Line Sensors: enabled with Tx

Pulse duration >= 10us