

VigiSense

11

Generated by Doxygen 1.10.0

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 eprosimas Namespace Reference	9
5.2 eprosimas::fastcdr Namespace Reference	9
5.3 tps Namespace Reference	9
5.3.1 Function Documentation	9
5.3.1.1 pow()	9
6 Class Documentation	11
6.1 alert Struct Reference	11
6.1.1 Detailed Description	12
6.1.2 Constructor & Destructor Documentation	12
6.1.2.1 alert() [1/3]	12
6.1.2.2 ~alert()	13
6.1.2.3 alert() [2/3]	13
6.1.2.4 alert() [3/3]	13
6.1.3 Member Function Documentation	13
6.1.3.1 deserialize()	13
6.1.3.2 getCdrSerializedSize()	13
6.1.3.3 getKeyMaxCdrSerializedSize()	14
6.1.3.4 getMaxCdrSerializedSize()	14
6.1.3.5 index() [1/3]	14
6.1.3.6 index() [2/3]	15
6.1.3.7 index() [3/3]	15
6.1.3.8 isKeyDefined()	15
6.1.3.9 message() [1/4]	15
6.1.3.10 message() [2/4]	15
6.1.3.11 message() [3/4]	16
6.1.3.12 message() [4/4]	16
6.1.3.13 operator!=(())	16
6.1.3.14 operator=() [1/2]	16
6.1.3.15 operator=() [2/2]	17
6.1.3.16 operator==(())	17

6.1.3.17 serialize()	17
6.1.3.18 serializeKey()	17
6.1.4 Member Data Documentation	18
6.1.4.1 index	18
6.1.4.2 m_index	18
6.1.4.3 m_message	18
6.1.4.4 message	18
6.2 alertPubSubType Class Reference	19
6.2.1 Detailed Description	20
6.2.2 Member Typedef Documentation	20
6.2.2.1 type	20
6.2.3 Constructor & Destructor Documentation	20
6.2.3.1 alertPubSubType()	20
6.2.3.2 ~alertPubSubType()	21
6.2.4 Member Function Documentation	21
6.2.4.1 createData()	21
6.2.4.2 deleteData()	21
6.2.4.3 deserialize()	21
6.2.4.4 getKey()	21
6.2.4.5 getSerializedSizeProvider()	22
6.2.4.6 serialize()	22
6.2.5 Member Data Documentation	22
6.2.5.1 m_keyBuffer	22
6.2.5.2 m_md5	22
6.3 CircularDelay< type, size > Class Template Reference	23
6.3.1 Detailed Description	24
6.3.2 Constructor & Destructor Documentation	24
6.3.2.1 CircularDelay()	24
6.3.3 Member Function Documentation	25
6.3.3.1 begin()	25
6.3.3.2 end()	25
6.3.3.3 get()	25
6.3.3.4 push()	26
6.3.3.5 rbegin()	26
6.3.3.6 rend()	26
6.3.4 Member Data Documentation	26
6.3.4.1 data	26
6.3.4.2 setIterator	26
6.4 CircularDelay< type, size >::const_iterator Class Reference	27
6.4.1 Member Typedef Documentation	27
6.4.1.1 difference_type	27
6.4.1.2 iterator_category	27

6.4.1.3 self_type	28
6.4.2 Constructor & Destructor Documentation	28
6.4.2.1 const_iterator() [1/2]	28
6.4.2.2 const_iterator() [2/2]	28
6.4.3 Member Function Documentation	28
6.4.3.1 operator!=(())	28
6.4.3.2 operator*()	28
6.4.3.3 operator++() [1/2]	28
6.4.3.4 operator++() [2/2]	28
6.4.3.5 operator--() [1/2]	29
6.4.3.6 operator--() [2/2]	29
6.4.3.7 operator->()	29
6.4.3.8 operator==(())	29
6.4.3.9 operator[]()	29
6.4.4 Friends And Related Symbol Documentation	29
6.4.4.1 CircularDelay	29
6.4.5 Member Data Documentation	29
6.4.5.1 data_	29
6.4.5.2 ptr_	30
6.5 CircularDelay< type, size >::const_reverse_iterator Class Reference	30
6.5.1 Member Typedef Documentation	31
6.5.1.1 difference_type	31
6.5.1.2 iterator_category	31
6.5.1.3 self_type	31
6.5.2 Constructor & Destructor Documentation	31
6.5.2.1 const_reverse_iterator() [1/2]	31
6.5.2.2 const_reverse_iterator() [2/2]	31
6.5.3 Member Function Documentation	31
6.5.3.1 operator!=(())	31
6.5.3.2 operator*()	31
6.5.3.3 operator++() [1/2]	32
6.5.3.4 operator++() [2/2]	32
6.5.3.5 operator--() [1/2]	32
6.5.3.6 operator--() [2/2]	32
6.5.3.7 operator->()	32
6.5.3.8 operator==(())	32
6.5.3.9 operator[]()	32
6.5.4 Friends And Related Symbol Documentation	32
6.5.4.1 CircularDelay	32
6.5.5 Member Data Documentation	33
6.5.5.1 data_	33
6.5.5.2 ptr_	33

6.6 contact Struct Reference	33
6.6.1 Member Data Documentation	33
6.6.1.1 email	33
6.6.1.2 name	33
6.6.1.3 phoneNum	33
6.7 DevicePublisher Class Reference	34
6.7.1 Constructor & Destructor Documentation	34
6.7.1.1 DevicePublisher()	34
6.7.1.2 ~DevicePublisher()	35
6.7.2 Member Function Documentation	35
6.7.2.1 init()	35
6.7.2.2 publish()	35
6.7.3 Member Data Documentation	35
6.7.3.1 listener_	35
6.7.3.2 participant_	35
6.7.3.3 publisher_	35
6.7.3.4 topic_	35
6.7.3.5 type_	35
6.7.3.6 writer_	36
6.8 DeviceSubscriber Class Reference	36
6.8.1 Constructor & Destructor Documentation	37
6.8.1.1 DeviceSubscriber()	37
6.8.1.2 ~DeviceSubscriber()	37
6.8.2 Member Function Documentation	37
6.8.2.1 init()	37
6.8.3 Member Data Documentation	37
6.8.3.1 listener_	37
6.8.3.2 participant_	37
6.8.3.3 reader_	37
6.8.3.4 subscriber_	37
6.8.3.5 topic_	37
6.8.3.6 type_	38
6.9 diagnosis::DiagnosesTable Struct Reference	38
6.9.1 Member Data Documentation	38
6.9.1.1 Diagnoses	38
6.10 diagnosis Class Reference	38
6.10.1 Member Typedef Documentation	39
6.10.1.1 _Diagnoses	39
6.10.1.2 Diagnosis_Range	39
6.10.2 Member Function Documentation	39
6.10.2.1 critCheck()	39
6.10.2.2 critRangeAlert()	39

6.10.2.3 determineDiagnosis()	40
6.10.2.4 displayDiagnosis()	40
6.10.2.5 findMinMax()	40
6.10.2.6 SetdiagnosisRanges()	40
6.10.3 Member Data Documentation	40
6.10.3.1 CriticalHigh	40
6.10.3.2 CriticalLow	40
6.10.3.3 CustomDiagnoses	40
6.10.3.4 CustomDiagnosis	40
6.10.3.5 stdDiagnoses	40
6.10.3.6 stdDiagnosis	41
6.11 diagnosisInterface Class Reference	41
6.11.1 Member Function Documentation	41
6.11.1.1 determineSymptom()	41
6.11.1.2 getVal()	42
6.11.1.3 ping()	42
6.11.1.4 start()	42
6.11.1.5 stop()	42
6.11.2 Member Data Documentation	42
6.11.2.1 symptomRanges	42
6.12 diagnosis::DiagnosisRange Struct Reference	42
6.12.1 Member Data Documentation	43
6.12.1.1 diagnosis	43
6.12.1.2 max	43
6.12.1.3 min	43
6.13 Differentiator< T > Class Template Reference	43
6.13.1 Detailed Description	44
6.13.2 Constructor & Destructor Documentation	44
6.13.2.1 Differentiator()	44
6.13.3 Member Function Documentation	44
6.13.3.1 getOutput()	44
6.13.3.2 update()	44
6.13.4 Member Data Documentation	45
6.13.4.1 sampleTime	45
6.13.4.2 x1	45
6.13.4.3 y	45
6.14 DigitalFilter< Type > Class Template Reference	45
6.14.1 Detailed Description	45
6.14.2 Member Function Documentation	46
6.14.2.1 getOutput()	46
6.14.2.2 update()	46
6.15 HighPassFilter Class Reference	46

6.15.1 Detailed Description	47
6.15.2 Constructor & Destructor Documentation	47
6.15.2.1 HighPassFilter()	47
6.15.3 Member Function Documentation	48
6.15.3.1 configOutput()	48
6.15.3.2 getOutput()	48
6.15.3.3 outputPointer()	48
6.15.3.4 update()	48
6.15.4 Member Data Documentation	49
6.15.4.1 amplFac	49
6.15.4.2 dt	49
6.15.4.3 output	49
6.15.4.4 x1	49
6.15.4.5 y1c	49
6.16 HighPassFilter3 Class Reference	50
6.16.1 Detailed Description	51
6.16.2 Constructor & Destructor Documentation	51
6.16.2.1 HighPassFilter3()	51
6.16.3 Member Function Documentation	51
6.16.3.1 getOutput()	51
6.16.3.2 update()	51
6.16.4 Member Data Documentation	52
6.16.4.1 x	52
6.16.4.2 xc	52
6.16.4.3 y	52
6.16.4.4 yc	52
6.17 HRTracker Class Reference	52
6.17.1 Constructor & Destructor Documentation	54
6.17.1.1 HRTracker()	54
6.17.1.2 ~HRTracker()	54
6.17.2 Member Function Documentation	54
6.17.2.1 getVal()	54
6.17.2.2 ping()	55
6.17.2.3 pingThread()	55
6.17.2.4 start()	55
6.17.2.5 stop()	56
6.17.2.6 tracker()	56
6.17.3 Member Data Documentation	56
6.17.3.1 _s	56
6.17.3.2 symptomRanges	56
6.17.3.3 threadRunning	56
6.18 i2c_msg Struct Reference	57

6.18.1 Member Data Documentation	57
6.18.1.1 addr	57
6.18.1.2 buf	57
6.18.1.3 flags	57
6.18.1.4 len	57
6.19 i2c_rdwir_ioctl_data Struct Reference	57
6.19.1 Member Data Documentation	58
6.19.1.1 msgs	58
6.19.1.2 nmsgs	58
6.20 i2c_smbus_data Union Reference	58
6.20.1 Member Data Documentation	58
6.20.1.1 block	58
6.20.1.2 byte	58
6.20.1.3 word	58
6.21 i2c_smbus_ioctl_data Struct Reference	59
6.21.1 Member Data Documentation	59
6.21.1.1 command	59
6.21.1.2 data	59
6.21.1.3 read_write	59
6.21.1.4 size	59
6.22 CircularDelay< type, size >::iterator Class Reference	60
6.22.1 Member Typedef Documentation	60
6.22.1.1 difference_type	60
6.22.1.2 iterator_category	60
6.22.1.3 self_type	61
6.22.2 Constructor & Destructor Documentation	61
6.22.2.1 iterator() [1/2]	61
6.22.2.2 iterator() [2/2]	61
6.22.3 Member Function Documentation	61
6.22.3.1 operator!=(())	61
6.22.3.2 operator*()	61
6.22.3.3 operator++() [1/2]	61
6.22.3.4 operator++() [2/2]	61
6.22.3.5 operator--() [1/2]	62
6.22.3.6 operator--() [2/2]	62
6.22.3.7 operator->()	62
6.22.3.8 operator==(())	62
6.22.3.9 operator[]()	62
6.22.4 Friends And Related Symbol Documentation	62
6.22.4.1 CircularDelay	62
6.22.5 Member Data Documentation	62
6.22.5.1 data_	62

6.22.5.2 ptr_	63
6.23 LowPassFilter Class Reference	63
6.23.1 Detailed Description	64
6.23.2 Constructor & Destructor Documentation	64
6.23.2.1 LowPassFilter()	64
6.23.3 Member Function Documentation	65
6.23.3.1 configOutput()	65
6.23.3.2 getOutput()	65
6.23.3.3 outputPointer()	65
6.23.3.4 update()	65
6.23.4 Member Data Documentation	66
6.23.4.1 dt	66
6.23.4.2 epow	66
6.23.4.3 output	66
6.24 LowPassFilter2 Class Reference	66
6.24.1 Detailed Description	67
6.24.2 Constructor & Destructor Documentation	67
6.24.2.1 LowPassFilter2()	67
6.24.3 Member Function Documentation	68
6.24.3.1 configOutput()	68
6.24.3.2 getOutput()	68
6.24.3.3 update()	68
6.24.4 Member Data Documentation	69
6.24.4.1 x	69
6.24.4.2 xc	69
6.24.4.3 y	69
6.24.4.4 yc	69
6.25 LowPassFilter3 Class Reference	70
6.25.1 Detailed Description	71
6.25.2 Constructor & Destructor Documentation	71
6.25.2.1 LowPassFilter3()	71
6.25.3 Member Function Documentation	71
6.25.3.1 getOutput()	71
6.25.3.2 update()	71
6.25.4 Member Data Documentation	72
6.25.4.1 x	72
6.25.4.2 xc	72
6.25.4.3 y	72
6.25.4.4 yc	72
6.26 LowPassFilter3DiffApprox Class Reference	72
6.26.1 Detailed Description	73
6.26.2 Constructor & Destructor Documentation	73

6.26.2.1 LowPassFilter3DiffApprox()	73
6.26.3 Member Function Documentation	74
6.26.3.1 getOutput()	74
6.26.3.2 update()	74
6.26.4 Member Data Documentation	74
6.26.4.1 x	74
6.26.4.2 xc	74
6.26.4.3 y	75
6.26.4.4 yc	75
6.27 LowPassFilter3MatchedZ Class Reference	75
6.27.1 Detailed Description	76
6.27.2 Constructor & Destructor Documentation	76
6.27.2.1 LowPassFilter3MatchedZ()	76
6.27.3 Member Function Documentation	76
6.27.3.1 getOutput()	76
6.27.3.2 update()	77
6.27.4 Member Data Documentation	77
6.27.4.1 amplFac	77
6.27.4.2 y	77
6.27.4.3 yc	77
6.28 MAX30102 Class Reference	78
6.28.1 Member Typedef Documentation	80
6.28.1.1 sense_struct	80
6.28.2 Constructor & Destructor Documentation	80
6.28.2.1 MAX30102()	80
6.28.3 Member Function Documentation	80
6.28.3.1 available()	80
6.28.3.2 begin()	80
6.28.3.3 bitMask()	81
6.28.3.4 check()	81
6.28.3.5 clearFIFO()	81
6.28.3.6 dataReady()	82
6.28.3.7 disableAFULL()	82
6.28.3.8 disableALCOVF()	82
6.28.3.9 disableDATARDY()	83
6.28.3.10 disableDIETEMPRDY()	83
6.28.3.11 disableFIFORollover()	83
6.28.3.12 disablePROXINT()	84
6.28.3.13 disableSlots()	84
6.28.3.14 enableAFULL()	84
6.28.3.15 enableALCOVF()	85
6.28.3.16 enableDATARDY()	85

6.28.3.17 enableDIETEMPRDY()	85
6.28.3.18 enableFIFORollover()	86
6.28.3.19 enablePROXINT()	86
6.28.3.20 enableSlot()	86
6.28.3.21 getFIFOIR()	87
6.28.3.22 getFIFORed()	87
6.28.3.23 getINT1()	87
6.28.3.24 getINT2()	87
6.28.3.25 getIR()	88
6.28.3.26 getReadPointer()	88
6.28.3.27 getRed()	88
6.28.3.28 getRevisionID()	88
6.28.3.29 getWritePointer()	88
6.28.3.30 gpioISR()	89
6.28.3.31 hasSample()	89
6.28.3.32 nextSample()	89
6.28.3.33 readMany()	90
6.28.3.34 readPartID()	90
6.28.3.35 readRevisionID()	90
6.28.3.36 readTemperature()	91
6.28.3.37 readTemperatureF()	91
6.28.3.38 safeCheck()	91
6.28.3.39 setADCRange()	92
6.28.3.40 setFIFOAlmostFull()	92
6.28.3.41 setFIFOAverage()	92
6.28.3.42 setLEDMode()	93
6.28.3.43 setProximityThreshold()	93
6.28.3.44 setPROXINTTHRESH()	93
6.28.3.45 setPulseAmplitudeIR()	94
6.28.3.46 setPulseAmplitudeProximity()	94
6.28.3.47 setPulseAmplitudeRed()	94
6.28.3.48 setPulseWidth()	95
6.28.3.49 setSampleRate()	95
6.28.3.50 setup()	95
6.28.3.51 shutDown()	96
6.28.3.52 softReset()	96
6.28.3.53 wakeUp()	97
6.28.4 Member Data Documentation	97
6.28.4.1 _i2c	97
6.28.4.2 _i2caddr	97
6.28.4.3 activeLEDs	97
6.28.4.4 revisionID	97

6.28.4.5 sense	98
6.29 MovingAverageFilter< size > Class Template Reference	98
6.29.1 Member Function Documentation	99
6.29.1.1 update()	99
6.29.2 Member Data Documentation	99
6.29.2.1 buffer	99
6.29.2.2 output	99
6.30 DevicePublisher::PubListener Class Reference	100
6.30.1 Constructor & Destructor Documentation	100
6.30.1.1 PubListener()	100
6.30.1.2 ~PubListener()	101
6.30.2 Member Function Documentation	101
6.30.2.1 on_publication_matched()	101
6.30.3 Member Data Documentation	101
6.30.3.1 matched_	101
6.31 MAX30102::Record Struct Reference	101
6.31.1 Member Data Documentation	101
6.31.1.1 head	101
6.31.1.2 IR	101
6.31.1.3 red	101
6.31.1.4 tail	102
6.32 CircularDelay< type, size >::reverse_iterator Class Reference	102
6.32.1 Member Typedef Documentation	103
6.32.1.1 difference_type	103
6.32.1.2 iterator_category	103
6.32.1.3 self_type	103
6.32.2 Constructor & Destructor Documentation	103
6.32.2.1 reverse_iterator() [1/2]	103
6.32.2.2 reverse_iterator() [2/2]	103
6.32.3 Member Function Documentation	103
6.32.3.1 operator!=(())	103
6.32.3.2 operator*()	103
6.32.3.3 operator++() [1/2]	104
6.32.3.4 operator++() [2/2]	104
6.32.3.5 operator--() [1/2]	104
6.32.3.6 operator--() [2/2]	104
6.32.3.7 operator->()	104
6.32.3.8 operator==(())	104
6.32.3.9 operator[]()	104
6.32.4 Friends And Related Symbol Documentation	104
6.32.4.1 CircularDelay	104
6.32.5 Member Data Documentation	105

6.32.5.1 data_	105
6.32.5.2 ptr_	105
6.33 sensor Class Reference	105
6.33.1 Constructor & Destructor Documentation	107
6.33.1.1 sensor()	107
6.33.1.2 ~sensor()	107
6.33.2 Member Function Documentation	107
6.33.2.1 begin()	107
6.33.2.2 Derivative()	108
6.33.2.3 getHR()	108
6.33.2.4 getLatestTemperatureF()	108
6.33.2.5 getPeakThreshold()	108
6.33.2.6 getSpO2()	108
6.33.2.7 HRcalc()	108
6.33.2.8 loopThread()	109
6.33.2.9 peakDetect()	109
6.33.2.10 resetCalculations()	109
6.33.2.11 runHRCalculationLoop()	110
6.33.2.12 stop()	110
6.33.2.13 stopHRcalc()	110
6.33.2.14 updateTemperature()	110
6.33.3 Member Data Documentation	111
6.33.3.1 _sensor	111
6.33.3.2 averageIRBPM	111
6.33.3.3 BPM_BUFFER_SIZE	111
6.33.3.4 bpmBuffer	111
6.33.3.5 irLastValue	111
6.33.3.6 latestIRBPM	111
6.33.3.7 latestRedBPM	111
6.33.3.8 latestSpO2	111
6.33.3.9 latestTemperature	111
6.33.3.10 localMaximalR	111
6.33.3.11 localMaximaRed	112
6.33.3.12 localMinimalR	112
6.33.3.13 localMinimaRed	112
6.33.3.14 nextBPMBufferIndex	112
6.33.3.15 nextSPO2BufferIndex	112
6.33.3.16 PAST_PEAKS_SIZE	112
6.33.3.17 pastMaximasIR	112
6.33.3.18 pastMaximasRed	112
6.33.3.19 pastMinimasIR	112
6.33.3.20 pastMinimasRed	112

6.33.3.21 R	113
6.33.3.22 redLastValue	113
6.33.3.23 running	113
6.33.3.24 runningHR	113
6.33.3.25 SPO2_BUFFER_SIZE	113
6.33.3.26 spo2Buffer	113
6.33.3.27 timeLastIRHeartBeat	113
6.33.3.28 timeLastLoopRan	113
6.33.3.29 timeLastRedHeartBeat	113
6.34 SPO2Tracker Class Reference	114
6.34.1 Constructor & Destructor Documentation	115
6.34.1.1 SPO2Tracker()	115
6.34.1.2 ~SPO2Tracker()	115
6.34.2 Member Function Documentation	116
6.34.2.1 getVal()	116
6.34.2.2 ping()	116
6.34.2.3 pingThread()	116
6.34.2.4 start()	117
6.34.2.5 stop()	117
6.34.2.6 tracker()	117
6.34.3 Member Data Documentation	117
6.34.3.1 _s	117
6.34.3.2 symptomRanges	118
6.34.3.3 threadRunning	118
6.35 DeviceSubscriber::SubListener Class Reference	118
6.35.1 Constructor & Destructor Documentation	119
6.35.1.1 SubListener()	119
6.35.1.2 ~SubListener()	119
6.35.2 Member Function Documentation	119
6.35.2.1 on_data_available()	119
6.35.2.2 on_subscription_matched()	119
6.36 symptomRange Struct Reference	119
6.36.1 Member Data Documentation	120
6.36.1.1 max	120
6.36.1.2 min	120
6.36.1.3 symptom	120
7 File Documentation	121
7.1 /home/sitcomlab/Projects/VigiSense/src/alert.cxx File Reference	121
7.2 /home/sitcomlab/Projects/VigiSense/src/alert.h File Reference	121
7.2.1 Detailed Description	122
7.2.2 Macro Definition Documentation	123

7.2.2.1 alert_DllAPI	123
7.2.2.2 eProxima_user_DllExport	123
7.3 alert.h	123
7.4 /home/sitcomlab/Projects/VigiSense/src/alert.idl File Reference	124
7.5 alert.idl	125
7.6 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx File Reference	125
7.6.1 Typedef Documentation	125
7.6.1.1 InstanceHandle_t	125
7.6.1.2 SerializedPayload_t	125
7.7 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h File Reference	126
7.7.1 Detailed Description	126
7.8 alertPubSubTypes.h	127
7.9 /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp File Reference	128
7.10 CircularDelay.hpp	129
7.11 /home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp File Reference	131
7.12 DevicePublisher.cpp	132
7.13 /home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp File Reference	134
7.14 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp File Reference	135
7.15 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h File Reference	135
7.16 Diagnosis.h	136
7.17 /home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h File Reference	137
7.18 DiagnosisInterface.h	138
7.19 /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h File Reference	138
7.19.1 Function Documentation	140
7.19.1.1 calcC_Cr()	140
7.19.1.2 squareOf()	140
7.20 DigitalFilters.h	141
7.21 /home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp File Reference	145
7.22 /home/sitcomlab/Projects/VigiSense/src/HRTracker.h File Reference	145
7.23 HRTracker.h	146
7.24 /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h File Reference	147
7.24.1 Macro Definition Documentation	149
7.24.1.1 I2C_FUNC_10BIT_ADDR	149
7.24.1.2 I2C_FUNC_I2C	149
7.24.1.3 I2C_FUNC_PROTOCOL_MANGLING	149
7.24.1.4 I2C_FUNC_SMBUS_BLOCK_DATA	149
7.24.1.5 I2C_FUNC_SMBUS_BLOCK_PROC_CALL	149
7.24.1.6 I2C_FUNC_SMBUS_BYTE	150
7.24.1.7 I2C_FUNC_SMBUS_BYTE_DATA	150
7.24.1.8 I2C_FUNC_SMBUS_HWPEC_CALC	150
7.24.1.9 I2C_FUNC_SMBUS_I2C_BLOCK	150
7.24.1.10 I2C_FUNC_SMBUS_PEC	150

7.24.1.11 I2C_FUNC_SMBUS_PROC_CALL	150
7.24.1.12 I2C_FUNC_SMBUS_QUICK	150
7.24.1.13 I2C_FUNC_SMBUS_READ_BLOCK_DATA	150
7.24.1.14 I2C_FUNC_SMBUS_READ_BYTE	151
7.24.1.15 I2C_FUNC_SMBUS_READ_BYTE_DATA	151
7.24.1.16 I2C_FUNC_SMBUS_READ_I2C_BLOCK	151
7.24.1.17 I2C_FUNC_SMBUS_READ_WORD_DATA	151
7.24.1.18 I2C_FUNC_SMBUS_WORD_DATA	151
7.24.1.19 I2C_FUNC_SMBUS_WRITE_BLOCK_DATA	151
7.24.1.20 I2C_FUNC_SMBUS_WRITE_BYTE	151
7.24.1.21 I2C_FUNC_SMBUS_WRITE_BYTE_DATA	151
7.24.1.22 I2C_FUNC_SMBUS_WRITE_I2C_BLOCK	151
7.24.1.23 I2C_FUNC_SMBUS_WRITE_WORD_DATA	151
7.24.1.24 I2C_FUNCS	152
7.24.1.25 I2C_M_IGNORE_NAK	152
7.24.1.26 I2C_M_NO_RD_ACK	152
7.24.1.27 I2C_M_NOSTART	152
7.24.1.28 I2C_M_RD	152
7.24.1.29 I2C_M_REV_DIR_ADDR	152
7.24.1.30 I2C_M_TEN	152
7.24.1.31 I2C_PEC	152
7.24.1.32 I2C_RDRW_IOCTL_MAX_MSGS	152
7.24.1.33 I2C_RDWR	152
7.24.1.34 I2C_RETRIES	153
7.24.1.35 I2C_SLAVE	153
7.24.1.36 I2C_SLAVE_FORCE	153
7.24.1.37 I2C_SMBUS	153
7.24.1.38 I2C_SMBUS_BLOCK_DATA	153
7.24.1.39 I2C_SMBUS_BLOCK_MAX	153
7.24.1.40 I2C_SMBUS_BLOCK_PROC_CALL	153
7.24.1.41 I2C_SMBUS_BYTE	153
7.24.1.42 I2C_SMBUS_BYTE_DATA	153
7.24.1.43 I2C_SMBUS_I2C_BLOCK_BROKEN	154
7.24.1.44 I2C_SMBUS_I2C_BLOCK_DATA	154
7.24.1.45 I2C_SMBUS_I2C_BLOCK_MAX	154
7.24.1.46 I2C_SMBUS_PROC_CALL	154
7.24.1.47 I2C_SMBUS_QUICK	154
7.24.1.48 I2C_SMBUS_READ	154
7.24.1.49 I2C_SMBUS_WORD_DATA	154
7.24.1.50 I2C_SMBUS_WRITE	154
7.24.1.51 I2C_TENBIT	154
7.24.1.52 I2C_TIMEOUT	154

7.24.2 Function Documentation	155
7.24.2.1 i2c_smbus_access()	155
7.24.2.2 i2c_smbus_block_process_call()	155
7.24.2.3 i2c_smbus_process_call()	155
7.24.2.4 i2c_smbus_read_block_data()	155
7.24.2.5 i2c_smbus_read_byte()	156
7.24.2.6 i2c_smbus_read_byte_data()	156
7.24.2.7 i2c_smbus_read_i2c_block_data()	156
7.24.2.8 i2c_smbus_read_word_data()	156
7.24.2.9 i2c_smbus_write_block_data()	156
7.24.2.10 i2c_smbus_write_byte()	157
7.24.2.11 i2c_smbus_write_byte_data()	157
7.24.2.12 i2c_smbus_write_i2c_block_data()	157
7.24.2.13 i2c_smbus_write_quick()	157
7.24.2.14 i2c_smbus_write_word_data()	157
7.25 i2c-dev.h	158
7.26 /home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp File Reference	161
7.26.1 Variable Documentation	164
7.26.1.1 ADCRANGE_16384	164
7.26.1.2 ADCRANGE_2048	164
7.26.1.3 ADCRANGE_4096	164
7.26.1.4 ADCRANGE_8192	164
7.26.1.5 INT_A_FULL_DISABLE	164
7.26.1.6 INT_A_FULL_ENABLE	164
7.26.1.7 INT_ALC_OVF_DISABLE	164
7.26.1.8 INT_ALC_OVF_ENABLE	164
7.26.1.9 INT_DATA_RDY_DISABLE	164
7.26.1.10 INT_DATA_RDY_ENABLE	164
7.26.1.11 INT_DIE_TEMP_RDY_DISABLE	165
7.26.1.12 INT_DIE_TEMP_RDY_ENABLE	165
7.26.1.13 INT_PROX_INT_DISABLE	165
7.26.1.14 INT_PROX_INT_ENABLE	165
7.26.1.15 LEDMODE_MULTILED	165
7.26.1.16 LEDMODE_REDIRONLY	165
7.26.1.17 LEDMODE_REDONLY	165
7.26.1.18 MASK_A_FULL	165
7.26.1.19 MASK_ADCRANGE	165
7.26.1.20 MASK_INT_A_FULL	165
7.26.1.21 MASK_INT_ALC_OVF	166
7.26.1.22 MASK_INT_DATA_RDY	166
7.26.1.23 MASK_INT_DIE_TEMP_RDY	166
7.26.1.24 MASK_INT_PROX_INT	166

7.26.1.25 MASK_LEDMODE	166
7.26.1.26 MASK_PULSEWIDTH	166
7.26.1.27 MASK_RESET	166
7.26.1.28 MASK_ROLLOVER	166
7.26.1.29 MASK_SAMPLEAVG	166
7.26.1.30 MASK_SAMPLERATE	166
7.26.1.31 MASK_SHUTDOWN	167
7.26.1.32 MASK_SLOT1	167
7.26.1.33 MASK_SLOT2	167
7.26.1.34 MASK_SLOT3	167
7.26.1.35 MASK_SLOT4	167
7.26.1.36 MAX30102_EXPECTEDPARTID	167
7.26.1.37 PULSEWIDTH_118	167
7.26.1.38 PULSEWIDTH_215	167
7.26.1.39 PULSEWIDTH_411	167
7.26.1.40 PULSEWIDTH_69	167
7.26.1.41 REG_DIETEMPCONFIG	168
7.26.1.42 REG_DIETEMPFRAC	168
7.26.1.43 REG_DIETEMPINT	168
7.26.1.44 REG_FIFOCONFIG	168
7.26.1.45 REG_FIFODATA	168
7.26.1.46 REG_FIFOOVERFLOW	168
7.26.1.47 REG_FIFOREADPTR	168
7.26.1.48 REG_FIFOWRITEPTR	168
7.26.1.49 REG_INTENABLE1	168
7.26.1.50 REG_INTENABLE2	168
7.26.1.51 REG_INTSTAT1	169
7.26.1.52 REG_INTSTAT2	169
7.26.1.53 REG_LED1_PULSEAMP	169
7.26.1.54 REG_LED2_PULSEAMP	169
7.26.1.55 REG_LED_PROX_AMP	169
7.26.1.56 REG_MODECONFIG	169
7.26.1.57 REG_MULTILEDCONFIG1	169
7.26.1.58 REG_MULTILEDCONFIG2	169
7.26.1.59 REG_PARTICLECONFIG	169
7.26.1.60 REG_PARTID	169
7.26.1.61 REG_PROXINTTHRESH	170
7.26.1.62 REG_REVISIONID	170
7.26.1.63 RESET	170
7.26.1.64 ROLLOVER_DISABLE	170
7.26.1.65 ROLLOVER_ENABLE	170
7.26.1.66 SAMPLEAVG_1	170

7.26.1.67 SAMPLEAVG_16	170
7.26.1.68 SAMPLEAVG_2	170
7.26.1.69 SAMPLEAVG_32	170
7.26.1.70 SAMPLEAVG_4	170
7.26.1.71 SAMPLEAVG_8	171
7.26.1.72 SAMPLERATE_100	171
7.26.1.73 SAMPLERATE_1000	171
7.26.1.74 SAMPLERATE_1600	171
7.26.1.75 SAMPLERATE_200	171
7.26.1.76 SAMPLERATE_3200	171
7.26.1.77 SAMPLERATE_400	171
7.26.1.78 SAMPLERATE_50	171
7.26.1.79 SAMPLERATE_800	171
7.26.1.80 SHUTDOWN	171
7.26.1.81 SLOT_IR_LED	172
7.26.1.82 SLOT_IR_PILOT	172
7.26.1.83 SLOT_NONE	172
7.26.1.84 SLOT_NONE_PILOT	172
7.26.1.85 SLOT_RED_LED	172
7.26.1.86 SLOT_RED_PILOT	172
7.26.1.87 WAKEUP	172
7.27 /home/sitcomlab/Projects/VigiSense/src/MAX30102.h File Reference	172
7.27.1 Macro Definition Documentation	173
7.27.1.1 DEFAULT_INT_GPIO	173
7.27.1.2 I2C_BUFFER_LENGTH	173
7.27.1.3 I2C_SPEED_FAST	173
7.27.1.4 I2C_SPEED_STANDARD	173
7.27.1.5 MAX30102_ADDRESS	173
7.27.1.6 STORAGE_SIZE	174
7.28 MAX30102.h	174
7.29 /home/sitcomlab/Projects/VigiSense/src/Sensor.cpp File Reference	175
7.29.1 Variable Documentation	176
7.29.1.1 crest	176
7.29.1.2 dataBeenIncreasing	176
7.29.1.3 hpf	176
7.29.1.4 lpf	176
7.29.1.5 nextPastPeaksIndex	177
7.29.1.6 R	177
7.29.1.7 SpO2	177
7.29.1.8 trough	177
7.30 /home/sitcomlab/Projects/VigiSense/src/Sensor.h File Reference	177
7.31 Sensor.h	178

7.32 /home/sitcomlab/Projects/VigiSense/src/shutdown.cpp File Reference	179
7.32.1 Function Documentation	180
7.32.1.1 main()	180
7.32.2 Variable Documentation	180
7.32.2.1 heartSensor	180
7.33 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp File Reference	180
7.34 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h File Reference	181
7.35 SPO2Tracker.h	181
7.36 /home/sitcomlab/Projects/VigiSense/src/test.cpp File Reference	182
7.36.1 Function Documentation	183
7.36.1.1 main()	183
7.36.2 Variable Documentation	183
7.36.2.1 heartSensor	183
7.37 /home/sitcomlab/Projects/VigiSense/src/testSpO2.cpp File Reference	183
7.37.1 Function Documentation	184
7.37.1.1 main()	184
7.38 /home/sitcomlab/Projects/VigiSense/src/User.cpp File Reference	185
7.38.1 Function Documentation	185
7.38.1.1 addContact()	185
7.38.1.2 User()	185
7.38.2 Variable Documentation	186
7.38.2.1 contacts	186
7.38.2.2 name	186
Index	187

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

eprosima	9
eprosima::fastcdr	9
tps	9

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

alert	11
CircularDelay< type, size >	23
CircularDelay< double, 2 >	23
CircularDelay< double, 3 >	23
CircularDelay< double, 4 >	23
CircularDelay< int64_t, size >	23
CircularDelay< type, size >::const_iterator	27
CircularDelay< type, size >::const_reverse_iterator	30
contact	33
DataReaderListener	
DeviceSubscriber::SubListener	118
DataWriterListener	
DevicePublisher::PubListener	100
DevicePublisher	34
DeviceSubscriber	36
diagnosis::DiagnosesTable	38
diagnosis	38
diagnosisInterface	41
HRTracker	52
SPO2Tracker	114
diagnosis::DiagnosisRange	42
DigitalFilter< Type >	45
DigitalFilter< double >	45
HighPassFilter	46
HighPassFilter3	50
LowPassFilter	63
LowPassFilter2	66
LowPassFilter3	70
LowPassFilter3DiffApprox	72
LowPassFilter3MatchedZ	75
DigitalFilter< T >	45
Differentiator< T >	43
i2c_msg	57
i2c_rdwr_ioctl_data	57

i2c_smbus_data	58
i2c_smbus_ioctl_data	59
CircularDelay< type, size >::iterator	60
MAX30102	78
MovingAvarageFilter< size >	98
MAX30102::Record	101
CircularDelay< type, size >::reverse_iterator	102
sensor	105
symptomRange	119
eprosima::fastdds::dds::TopicDataType	
alertPubSubType	19

Chapter 3

Class Index

3.1 Class List

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

alert	This class represents the structure alert defined by the user in the IDL file	11
alertPubSubType	This class represents the TopicDataType of the type alert defined by the user in the IDL file . .	19
CircularDelay< type, size >	A class that functions as a sample buffer	23
CircularDelay< type, size >::const_iterator		27
CircularDelay< type, size >::const_reverse_iterator		30
contact		33
DevicePublisher		34
DeviceSubscriber		36
diagnosis::DiagnosesTable		38
diagnosis		38
diagnosisInterface		41
diagnosis::DiagnosisRange		42
Differentiator< T >	Class for differentiator	43
DigitalFilter< Type >	Abstract base class for digital moving filters	45
HighPassFilter	Class for high pass filter using bilinear transform	46
HighPassFilter3	Class for third order high pass filter. This is designed using the bilinear transform	50
HRTracker		52
i2c_msg		57
i2c_rdwr_ioctl_data		57
i2c_smbus_data		58
i2c_smbus_ioctl_data		59
CircularDelay< type, size >::iterator		60
LowPassFilter	Class for a low pass filter	63
LowPassFilter2	Class for a 2nd order low pass filter	66
LowPassFilter3	Class for third order high pass filter. This is designed using the bilinear transform	70

LowPassFilter3DiffApprox	
Class for third order high pass filter. This is designed using the approximated differtial approuch where $s=(Z-1)/(Z*T)$	72
LowPassFilter3MatchedZ	
Class for third order high pass filter. This is designed using the matched Z transform	75
MAX30102	78
MovingAvarageFilter< size >	98
DevicePublisher::PubListener	100
MAX30102::Record	101
CircularDelay< type, size >::reverse_iterator	102
sensor	105
SPO2Tracker	114
DeviceSubscriber::SubListener	118
symptomRange	119

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

/home/sitcomlab/Projects/VigiSense/src/alert.cxx	121
/home/sitcomlab/Projects/VigiSense/src/alert.h	121
/home/sitcomlab/Projects/VigiSense/src/alert.idl	124
/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx	125
/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h	126
/home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp	128
/home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp	131
/home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp	134
/home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp	135
/home/sitcomlab/Projects/VigiSense/src/Diagnosis.h	135
/home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h	137
/home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h	138
/home/sitcomlab/Projects/VigiSense/src/HRTTracker.cpp	145
/home/sitcomlab/Projects/VigiSense/src/HRTTracker.h	145
/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h	147
/home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp	161
/home/sitcomlab/Projects/VigiSense/src/MAX30102.h	172
/home/sitcomlab/Projects/VigiSense/src/Sensor.cpp	175
/home/sitcomlab/Projects/VigiSense/src/Sensor.h	177
/home/sitcomlab/Projects/VigiSense/src/shutdown.cpp	179
/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp	180
/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h	181
/home/sitcomlab/Projects/VigiSense/src/test.cpp	182
/home/sitcomlab/Projects/VigiSense/src/testSpO2.cpp	183
/home/sitcomlab/Projects/VigiSense/src/User.cpp	185

Chapter 5

Namespace Documentation

5.1 eprosima Namespace Reference

Namespaces

- namespace [fastcdr](#)

5.2 eprosima::fastcdr Namespace Reference

5.3 tps Namespace Reference

Functions

- `template<typename T >`
`constexpr T pow (T input, unsigned int power)`

5.3.1 Function Documentation

5.3.1.1 `pow()`

```
template<typename T >
constexpr T tps::pow (
    T input,
    unsigned int power ) [constexpr]
```

Here is the call graph for this function:



Chapter 6

Class Documentation

6.1 alert Struct Reference

This class represents the structure alert defined by the user in the IDL file.

```
#include <alert.h>
```

Public Member Functions

- [eProsima_user_DllExport alert \(\)](#)
Default constructor.
- [eProsima_user_DllExport ~alert \(\)](#)
Default destructor.
- [eProsima_user_DllExport alert \(const alert &x\)](#)
Copy constructor.
- [eProsima_user_DllExport alert \(alert &&x\) noexcept](#)
Move constructor.
- [eProsima_user_DllExport alert & operator= \(const alert &x\)](#)
Copy assignment.
- [eProsima_user_DllExport alert & operator= \(alert &&x\) noexcept](#)
Move assignment.
- [eProsima_user_DllExport bool operator== \(const alert &x\) const](#)
Comparison operator.
- [eProsima_user_DllExport bool operator!= \(const alert &x\) const](#)
Comparison operator.
- [eProsima_user_DllExport void index \(uint32_t _index\)](#)
This function sets a value in member index.
- [eProsima_user_DllExport uint32_t index \(\) const](#)
This function returns the value of member index.
- [eProsima_user_DllExport uint32_t & index \(\)](#)
This function returns a reference to member index.
- [eProsima_user_DllExport void message \(const std::string &_message\)](#)
This function copies the value in member message.
- [eProsima_user_DllExport void message \(std::string &&_message\)](#)
This function moves the value in member message.

- `eProsima_user_DllExport` `const std::string & message () const`
This function returns a constant reference to member message.
- `eProsima_user_DllExport` `std::string & message ()`
This function returns a reference to member message.
- `eProsima_user_DllExport` `void serialize (eprosima::fastcdr::Cdr &cdr) const`
This function serializes an object using CDR serialization.
- `eProsima_user_DllExport` `void deserialize (eprosima::fastcdr::Cdr &cdr)`
This function deserializes an object using CDR serialization.
- `eProsima_user_DllExport` `void serializeKey (eprosima::fastcdr::Cdr &cdr) const`
This function serializes the key members of an object using CDR serialization.

Static Public Member Functions

- static `eProsima_user_DllExport` `size_t getMaxCdrSerializedSize (size_t current_alignment=0)`
This function returns the maximum serialized size of an object depending on the buffer alignment.
- static `eProsima_user_DllExport` `size_t getCdrSerializedSize (const alert &data, size_t current_alignment=0)`
This function returns the serialized size of a data depending on the buffer alignment.
- static `eProsima_user_DllExport` `size_t getKeyMaxCdrSerializedSize (size_t current_alignment=0)`
This function returns the maximum serialized size of the Key of an object depending on the buffer alignment.
- static `eProsima_user_DllExport` `bool isKeyDefined ()`
This function tells you if the Key has been defined for this type.

Public Attributes

- unsigned long `index`
This function sets a value in member index.
- string `message`
This function copies the value in member message.

Private Attributes

- uint32_t `m_index`
- std::string `m_message`

6.1.1 Detailed Description

This class represents the structure alert defined by the user in the IDL file.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 alert() [1/3]

```
alert::alert ( )
```

Default constructor.

6.1.2.2 ~alert()

```
alert::~~alert ( )
```

Default destructor.

6.1.2.3 alert() [2/3]

```
alert::alert (
    const alert & x )
```

Copy constructor.

Parameters

<i>x</i>	Reference to the object alert that will be copied.
----------	--

6.1.2.4 alert() [3/3]

```
alert::alert (
    alert && x ) [noexcept]
```

Move constructor.

Parameters

<i>x</i>	Reference to the object alert that will be copied.
----------	--

6.1.3 Member Function Documentation

6.1.3.1 deserialize()

```
void alert::deserialize (
    eprosima::fastcdr::Cdr & cdr )
```

This function deserializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.1.3.2 getCdrSerializedSize()

```
size_t alert::getCdrSerializedSize (
    const alert & data,
    size_t current_alignment = 0 ) [static]
```

This function returns the serialized size of a data depending on the buffer alignment.

Parameters

<i>data</i>	Data which is calculated its serialized size.
<i>current_alignment</i>	Buffer alignment.

Returns

Serialized size.

6.1.3.3 getKeyMaxCdrSerializedSize()

```
size_t alert::getKeyMaxCdrSerializedSize (
    size_t current_alignment = 0 ) [static]
```

This function returns the maximum serialized size of the Key of an object depending on the buffer alignment.

Parameters

<i>current_alignment</i>	Buffer alignment.
--------------------------	-------------------

Returns

Maximum serialized size.

6.1.3.4 getMaxCdrSerializedSize()

```
size_t alert::getMaxCdrSerializedSize (
    size_t current_alignment = 0 ) [static]
```

This function returns the maximum serialized size of an object depending on the buffer alignment.

Parameters

<i>current_alignment</i>	Buffer alignment.
--------------------------	-------------------

Returns

Maximum serialized size.

6.1.3.5 index() [1/3]

```
eProsima_user_DllExport uint32_t & alert::index ( )
```

This function returns a reference to member index.

Returns

Reference to member index

6.1.3.6 index() [2/3]

```
eProsima_user_DllExport uint32_t alert::index ( ) const
```

This function returns the value of member index.

Returns

Value of member index

6.1.3.7 index() [3/3]

```
eProsima_user_DllExport void alert::index (
    uint32_t _index )
```

This function sets a value in member index.

Parameters

<code>_index</code>	New value for member index
---------------------	----------------------------

6.1.3.8 isKeyDefined()

```
bool alert::isKeyDefined ( ) [static]
```

This function tells you if the Key has been defined for this type.

6.1.3.9 message() [1/4]

```
eProsima_user_DllExport std::string & alert::message ( )
```

This function returns a reference to member message.

Returns

Reference to member message

6.1.3.10 message() [2/4]

```
eProsima_user_DllExport const std::string & alert::message ( ) const
```

This function returns a constant reference to member message.

Returns

Constant reference to member message

6.1.3.11 message() [3/4]

```
eProsima_user_DllExport void alert::message (
    const std::string & _message )
```

This function copies the value in member message.

Parameters

<code>_message</code>	New value to be copied in member message
-----------------------	--

6.1.3.12 message() [4/4]

```
eProsima_user_DllExport void alert::message (
    std::string && _message )
```

This function moves the value in member message.

Parameters

<code>_message</code>	New value to be moved in member message
-----------------------	---

6.1.3.13 operator"!="()

```
bool alert::operator!= (
    const alert & x ) const
```

Comparison operator.

Parameters

<code>x</code>	alert object to compare.
----------------	--------------------------

6.1.3.14 operator=() [1/2]

```
alert & alert::operator= (
    alert && x ) [noexcept]
```

Move assignment.

Parameters

<code>x</code>	Reference to the object alert that will be copied.
----------------	--

6.1.3.15 operator=() [2/2]

```
alert & alert::operator= (
    const alert & x )
```

Copy assignment.

Parameters

<i>x</i>	Reference to the object alert that will be copied.
----------	--

6.1.3.16 operator==()

```
bool alert::operator== (
    const alert & x ) const
```

Comparison operator.

Parameters

<i>x</i>	alert object to compare.
----------	--------------------------

6.1.3.17 serialize()

```
void alert::serialize (
    eprosima::fastcdr::Cdr & cdr ) const
```

This function serializes an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.1.3.18 serializeKey()

```
void alert::serializeKey (
    eprosima::fastcdr::Cdr & cdr ) const
```

This function serializes the key members of an object using CDR serialization.

Parameters

<i>cdr</i>	CDR serialization object.
------------	---------------------------

6.1.4 Member Data Documentation

6.1.4.1 index

```
uint32_t & alert::index
```

This function sets a value in member index.

This function returns a reference to member index.

This function returns the value of member index.

Parameters

<code>_index</code>	New value for member index
---------------------	----------------------------

Returns

Value of member index

Reference to member index

6.1.4.2 m_index

```
uint32_t alert::m_index [private]
```

6.1.4.3 m_message

```
std::string alert::m_message [private]
```

6.1.4.4 message

```
std::string & alert::message
```

This function copies the value in member message.

This function returns a reference to member message.

This function returns a constant reference to member message.

This function moves the value in member message.

Parameters

<code>_message</code>	New value to be copied in member message
<code>_message</code>	New value to be moved in member message

Returns

Constant reference to member message
Reference to member message

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

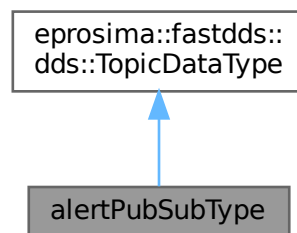
- [/home/sitcomlab/Projects/VigiSense/src/alert.h](#)
- [/home/sitcomlab/Projects/VigiSense/src/alert.idl](#)
- [/home/sitcomlab/Projects/VigiSense/src/alert.cxx](#)

6.2 alertPubSubType Class Reference

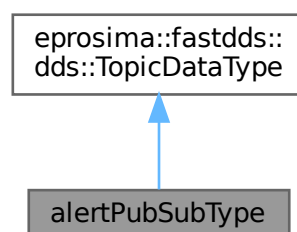
This class represents the TopicDataType of the type alert defined by the user in the IDL file.

```
#include <alertPubSubTypes.h>
```

Inheritance diagram for alertPubSubType:



Collaboration diagram for alertPubSubType:



Public Types

- typedef [alert type](#)

Public Member Functions

- [eProsima_user_DllExport alertPubSubType](#) ()
- virtual [eProsima_user_DllExport ~alertPubSubType](#) () override
- virtual [eProsima_user_DllExport](#) bool [serialize](#) (void *data, eprosima::fastrtps::rtps::SerializedPayload_↔ t *payload) override
- virtual [eProsima_user_DllExport](#) bool [deserialize](#) (eprosima::fastrtps::rtps::SerializedPayload_t *payload, void *data) override
- virtual [eProsima_user_DllExport](#) std::function< uint32_t()> [getSerializedSizeProvider](#) (void *data) override
- virtual [eProsima_user_DllExport](#) bool [getKey](#) (void *data, eprosima::fastrtps::rtps::InstanceHandle_↔ t *ihandle, bool force_md5=false) override
- virtual [eProsima_user_DllExport](#) void * [createData](#) () override
- virtual [eProsima_user_DllExport](#) void [deleteData](#) (void *data) override

Public Attributes

- MD5 [m_md5](#)
- unsigned char * [m_keyBuffer](#)

6.2.1 Detailed Description

This class represents the TopicDataType of the type alert defined by the user in the IDL file.

6.2.2 Member Typedef Documentation

6.2.2.1 type

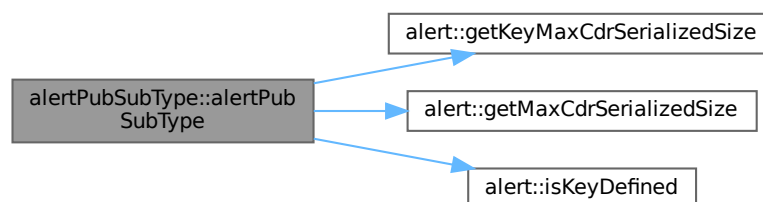
```
typedef alert alertPubSubType::type
```

6.2.3 Constructor & Destructor Documentation

6.2.3.1 alertPubSubType()

```
alertPubSubType::alertPubSubType ( )
```

Here is the call graph for this function:



6.2.3.2 ~alertPubSubType()

```
alertPubSubType::~~alertPubSubType ( ) [override], [virtual]
```

6.2.4 Member Function Documentation

6.2.4.1 createData()

```
void * alertPubSubType::createData ( ) [override], [virtual]
```

6.2.4.2 deleteData()

```
void alertPubSubType::deleteData (
    void * data ) [override], [virtual]
```

6.2.4.3 deserialize()

```
bool alertPubSubType::deserialize (
    eprosima::fastrtps::rtps::SerializedPayload_t * payload,
    void * data ) [override], [virtual]
```

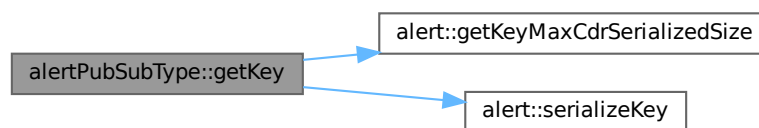
Here is the call graph for this function:



6.2.4.4 getKey()

```
bool alertPubSubType::getKey (
    void * data,
    eprosima::fastrtps::rtps::InstanceHandle_t * ihandle,
    bool force_md5 = false ) [override], [virtual]
```

Here is the call graph for this function:



6.2.4.5 getSerializedSizeProvider()

```
std::function< uint32_t()> alertPubSubType::getSerializedSizeProvider (
    void * data ) [override], [virtual]
```

Here is the call graph for this function:



6.2.4.6 serialize()

```
bool alertPubSubType::serialize (
    void * data,
    eprosima::fastrtps::rtps::SerializedPayload_t * payload ) [override], [virtual]
```

Here is the call graph for this function:



6.2.5 Member Data Documentation

6.2.5.1 m_keyBuffer

```
unsigned char* alertPubSubType::m_keyBuffer
```

6.2.5.2 m_md5

```
MD5 alertPubSubType::m_md5
```

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

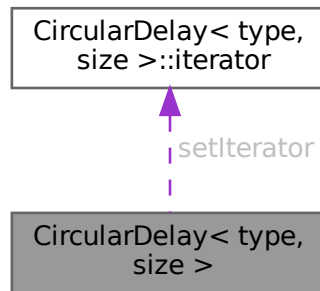
- [/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h](#)
- [/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx](#)

6.3 CircularDelay< type, size > Class Template Reference

A class that functions as a sample buffer.

```
#include <CircularDelay.hpp>
```

Collaboration diagram for CircularDelay< type, size >:



Classes

- class `const_iterator`
- class `const_reverse_iterator`
- class `iterator`
- class `reverse_iterator`

Public Member Functions

- `CircularDelay ()`
Constructor that initializes that buffer and its set index.
- type `push` (type input)
With this function you can insert a new sample into the buffer.
- type `get` (size_t delay)
With this function you can retrieve a sample from the past.
- `iterator end ()`
- `iterator begin ()`
- `reverse_iterator rend ()`
- `reverse_iterator rbegin ()`

Private Attributes

- type `data` [size+1]
- `iterator setIterator = iterator(data, data)`

6.3.1 Detailed Description

```
template<typename type, size_t size>
class CircularDelay< type, size >
```

A class that functions as a sample buffer.

Copyright

GPL V3

Circular delay software library. Here data can be stored and retrieved is a LiFo manner. Copyright (C) 2018 Jimmy van den Berg

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

You can use this to insert samples and use the get function to get a sample from the past.

Template Parameters

<i>type</i>	Type of sample that needs to be stored.
<i>size</i>	Size of how big the history buffer is.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 CircularDelay()

```
template<typename type , size_t size>
CircularDelay< type, size >::CircularDelay ( )
```

Constructor that initializes that buffer and its set index.

Template Parameters

<i>type</i>	Type of sample that needs to be stored.
<i>size</i>	Size of how big the history buffer is.

6.3.3 Member Function Documentation

6.3.3.1 begin()

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::begin ( ) [inline]
```

6.3.3.2 end()

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::end ( ) [inline]
```

6.3.3.3 get()

```
template<typename type , size_t size>
type CircularDelay< type, size >::get (
    size_t delay )
```

With this function you can retrieve a sample from the past.

Maximum delay is the size of the [CircularDelay](#) - 1.

Parameters

<i>delay</i>	How many samples you ago you want to get.
--------------	---

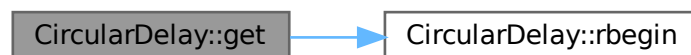
Template Parameters

<i>type</i>	Type of sample that needs to be stored.
<i>size</i>	Size of how big the history buffer is.

Returns

The sample of delay ago.

Here is the call graph for this function:



6.3.3.4 push()

```
template<typename type , size_t size>
type CircularDelay< type, size >::push (
    type input )
```

With this function you can insert a new sample into the buffer.

Parameters

<i>input</i>	Sample to push into.
--------------	----------------------

Template Parameters

<i>type</i>	Type of sample that needs to be stored.
<i>size</i>	Size of how big the history buffer is.

Returns

Value that has been pushed.

6.3.3.5 rbegin()

```
template<typename type , size_t size>
reverse_iterator CircularDelay< type, size >::rbegin ( ) [inline]
```

6.3.3.6 rend()

```
template<typename type , size_t size>
reverse_iterator CircularDelay< type, size >::rend ( ) [inline]
```

6.3.4 Member Data Documentation

6.3.4.1 data

```
template<typename type , size_t size>
type CircularDelay< type, size >::data[size+1] [private]
```

6.3.4.2 setIterator

```
template<typename type , size_t size>
iterator CircularDelay< type, size >::setIterator = iterator(data, data) [private]
```

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

- /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp

6.4 CircularDelay< type, size >::const_iterator Class Reference

```
#include <CircularDelay.hpp>
```

Public Types

- typedef [const_iterator](#) [self_type](#)
- typedef std::bidirectional_iterator_tag [iterator_category](#)
- typedef int [difference_type](#)

Public Member Functions

- [const_iterator](#) (const [CircularDelay](#)< type, size >::const_iterator &it)
- [self_type](#) operator++ ()
- [self_type](#) operator++ (int)
- [self_type](#) operator-- ()
- [self_type](#) operator-- (int)
- const type & [operator*](#) ()
- const type * [operator->](#) ()
- const type & [operator\[\]](#) (unsigned int index)
- bool [operator==](#) (const [self_type](#) &rhs)
- bool [operator!=](#) (const [self_type](#) &rhs)

Private Member Functions

- [const_iterator](#) (type *[data](#), type *ptr)

Private Attributes

- type * [data_](#) = nullptr
- type * [ptr_](#) = nullptr

Friends

- class [CircularDelay](#)

6.4.1 Member Typedef Documentation

6.4.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::const_iterator::difference_type
```

6.4.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::const_iterator::iterator↔
_category
```

6.4.1.3 self_type

```
template<typename type , size_t size>
typedef const\_iterator CircularDelay< type, size >::const_iterator::self_type
```

6.4.2 Constructor & Destructor Documentation

6.4.2.1 const_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_iterator::const_iterator (
    const CircularDelay< type, size >::const_iterator & it ) [inline]
```

6.4.2.2 const_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_iterator::const_iterator (
    type * data,
    type * ptr ) [inline], [private]
```

6.4.3 Member Function Documentation

6.4.3.1 operator"!="()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_iterator::operator!= (
    const self\_type & rhs ) [inline]
```

6.4.3.2 operator*()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_iterator::operator* ( ) [inline]
```

6.4.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self\_type CircularDelay< type, size >::const_iterator::operator++ ( ) [inline]
```

6.4.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self\_type CircularDelay< type, size >::const_iterator::operator++ (
    int ) [inline]
```

6.4.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator-- ( ) [inline]
```

6.4.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_iterator::operator-- (
    int ) [inline]
```

6.4.3.7 operator->()

```
template<typename type , size_t size>
const type * CircularDelay< type, size >::const_iterator::operator-> ( ) [inline]
```

6.4.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_iterator::operator== (
    const self_type & rhs ) [inline]
```

6.4.3.9 operator[]()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_iterator::operator[] (
    unsigned int index ) [inline]
```

6.4.4 Friends And Related Symbol Documentation

6.4.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay [friend]
```

6.4.5 Member Data Documentation

6.4.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_iterator::data_ = nullptr [private]
```

6.4.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_iterator::ptr_ = nullptr [private]
```

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

- </home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp>

6.5 CircularDelay< type, size >::const_reverse_iterator Class Reference

```
#include <CircularDelay.hpp>
```

Public Types

- typedef [const_reverse_iterator](#) [self_type](#)
- typedef std::bidirectional_iterator_tag [iterator_category](#)
- typedef int [difference_type](#)

Public Member Functions

- [const_reverse_iterator](#) (const [CircularDelay](#)< type, size >::const_reverse_iterator &it)
- [self_type](#) operator++ ()
- [self_type](#) operator++ (int)
- [self_type](#) operator-- ()
- [self_type](#) operator-- (int)
- const type & [operator*](#) ()
- const type * [operator->](#) ()
- const type & [operator\[\]](#) (int index)
- bool [operator==](#) (const [self_type](#) &rhs)
- bool [operator!=](#) (const [self_type](#) &rhs)

Private Member Functions

- [const_reverse_iterator](#) (type *data, type *ptr)

Private Attributes

- type * [data_](#) = nullptr
- type * [ptr_](#) = nullptr

Friends

- class [CircularDelay](#)

6.5.1 Member Typedef Documentation

6.5.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::const_reverse_iterator::difference_type
```

6.5.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::const_reverse_iterator↵
::iterator_category
```

6.5.1.3 self_type

```
template<typename type , size_t size>
typedef const_reverse_iterator CircularDelay< type, size >::const_reverse_iterator::self_type
```

6.5.2 Constructor & Destructor Documentation

6.5.2.1 const_reverse_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_reverse_iterator::const_reverse_iterator (
    const CircularDelay< type, size >::const_reverse_iterator & it ) [inline]
```

6.5.2.2 const_reverse_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::const_reverse_iterator::const_reverse_iterator (
    type * data,
    type * ptr ) [inline], [private]
```

6.5.3 Member Function Documentation

6.5.3.1 operator!=(())

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_reverse_iterator::operator!= (
    const self_type & rhs ) [inline]
```

6.5.3.2 operator*()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_reverse_iterator::operator* ( ) [inline]
```

6.5.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator++ ( ) [inline]
```

6.5.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator++ (
    int ) [inline]
```

6.5.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator-- ( ) [inline]
```

6.5.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::const_reverse_iterator::operator-- (
    int ) [inline]
```

6.5.3.7 operator->()

```
template<typename type , size_t size>
const type * CircularDelay< type, size >::const_reverse_iterator::operator-> ( ) [inline]
```

6.5.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::const_reverse_iterator::operator==(
    const self_type & rhs ) [inline]
```

6.5.3.9 operator[]()

```
template<typename type , size_t size>
const type & CircularDelay< type, size >::const_reverse_iterator::operator[] (
    int index ) [inline]
```

6.5.4 Friends And Related Symbol Documentation

6.5.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay [friend]
```

6.5.5 Member Data Documentation

6.5.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_reverse_iterator::data_ = nullptr [private]
```

6.5.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::const_reverse_iterator::ptr_ = nullptr [private]
```

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

- </home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp>

6.6 contact Struct Reference

Public Attributes

- string [name](#)
- string [email](#)
- long long [phoneNum](#)

6.6.1 Member Data Documentation

6.6.1.1 email

```
string contact::email
```

6.6.1.2 name

```
string contact::name
```

6.6.1.3 phoneNum

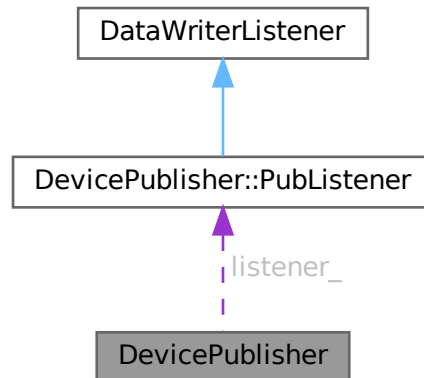
```
long long contact::phoneNum
```

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

- </home/sitcomlab/Projects/VigiSense/src/User.cpp>

6.7 DevicePublisher Class Reference

Collaboration diagram for DevicePublisher:



Classes

- class [PubListener](#)

Public Member Functions

- [DevicePublisher](#) ()
- virtual [~DevicePublisher](#) ()
- bool [init](#) ()
Initialize the publisher.
- bool [publish](#) ([alert](#) &hello)
Send a publication.

Private Attributes

- DomainParticipant * [participant_](#) = nullptr
- Publisher * [publisher_](#) = nullptr
- Topic * [topic_](#) = nullptr
- DataWriter * [writer_](#) = nullptr
- TypeSupport [type_](#)
- [DevicePublisher::PubListener](#) [listener_](#)

6.7.1 Constructor & Destructor Documentation

6.7.1.1 DevicePublisher()

```
DevicePublisher::DevicePublisher ( ) [inline]
```


6.7.1.2 ~DevicePublisher()

```
virtual DevicePublisher::~~DevicePublisher ( ) [inline], [virtual]
```

6.7.2 Member Function Documentation

6.7.2.1 init()

```
bool DevicePublisher::init ( ) [inline]
```

Initialize the publisher.

6.7.2.2 publish()

```
bool DevicePublisher::publish (
    alert & hello ) [inline]
```

Send a publication.

6.7.3 Member Data Documentation

6.7.3.1 listener_

```
DevicePublisher::PubListener DevicePublisher::listener_ [private]
```

6.7.3.2 participant_

```
DomainParticipant* DevicePublisher::participant_ = nullptr [private]
```

6.7.3.3 publisher_

```
Publisher* DevicePublisher::publisher_ = nullptr [private]
```

6.7.3.4 topic_

```
Topic* DevicePublisher::topic_ = nullptr [private]
```

6.7.3.5 type_

```
TypeSupport DevicePublisher::type_ [private]
```

6.7.3.6 writer_

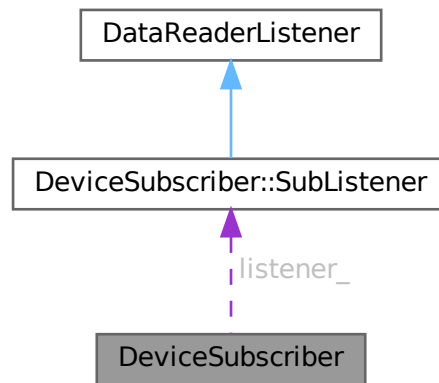
```
DataWriter* DevicePublisher::writer_ = nullptr [private]
```

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

- </home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp>

6.8 DeviceSubscriber Class Reference

Collaboration diagram for DeviceSubscriber:



Classes

- class [SubListener](#)

Public Member Functions

- [DeviceSubscriber](#) ()
- virtual [~DeviceSubscriber](#) ()
- bool [init](#) ()

Initialize the subscriber.

Private Attributes

- DomainParticipant * [participant_](#) = nullptr
- Subscriber * [subscriber_](#) = nullptr
- DataReader * [reader_](#) = nullptr
- Topic * [topic_](#) = nullptr
- TypeSupport [type_](#)
- [DeviceSubscriber::SubListener](#) [listener_](#)

6.8.1 Constructor & Destructor Documentation

6.8.1.1 DeviceSubscriber()

```
DeviceSubscriber::DeviceSubscriber ( ) [inline]
```

6.8.1.2 ~DeviceSubscriber()

```
virtual DeviceSubscriber::~~DeviceSubscriber ( ) [inline], [virtual]
```

6.8.2 Member Function Documentation

6.8.2.1 init()

```
bool DeviceSubscriber::init ( ) [inline]
```

Initialize the subscriber.

6.8.3 Member Data Documentation

6.8.3.1 listener_

```
DeviceSubscriber::SubListener DeviceSubscriber::listener_ [private]
```

6.8.3.2 participant_

```
DomainParticipant* DeviceSubscriber::participant_ = nullptr [private]
```

6.8.3.3 reader_

```
DataReader* DeviceSubscriber::reader_ = nullptr [private]
```

6.8.3.4 subscriber_

```
Subscriber* DeviceSubscriber::subscriber_ = nullptr [private]
```

6.8.3.5 topic_

```
Topic* DeviceSubscriber::topic_ = nullptr [private]
```

6.8.3.6 type_

```
TypeSupport DeviceSubscriber::type_ [private]
```

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

- [/home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp](#)

6.9 diagnosis::DiagnosesTable Struct Reference

```
#include <Diagnosis.h>
```

Public Attributes

- `std::vector< DiagnosisRange > Diagnoses`

6.9.1 Member Data Documentation

6.9.1.1 Diagnoses

```
std::vector<DiagnosisRange> diagnosis::DiagnosesTable::Diagnoses
```

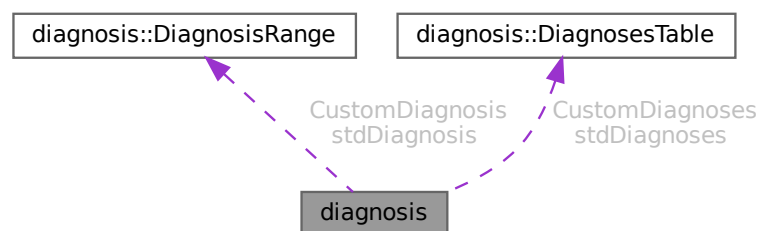
The documentation for this struct was generated from the following file:

- [/home/sitcomlab/Projects/VigiSense/src/Diagnosis.h](#)

6.10 diagnosis Class Reference

```
#include <Diagnosis.h>
```

Collaboration diagram for diagnosis:



Classes

- struct [DiagnosesTable](#)
- struct [DiagnosisRange](#)

Public Types

- typedef struct [diagnosis::DiagnosisRange](#) [Diagnosis_Range](#)
- typedef struct [diagnosis::DiagnosesTable](#) [_Diagnoses](#)

Public Member Functions

- void [SetdiagnosisRanges](#) (float minimum, float maximum, std::string [diagnosis](#))
- std::string [determineDiagnosis](#) ()
- void [critCheck](#) ()
- void [findMinMax](#) ()
- virtual void [displayDiagnosis](#) ()
- virtual void [critRangeAlert](#) ()

Public Attributes

- [Diagnosis_Range](#) stdDiagnosis
- [Diagnosis_Range](#) CustomDiagnosis
- [_Diagnoses](#) stdDiagnoses
- [_Diagnoses](#) CustomDiagnoses
- float [CriticalLow](#)
- float [CriticalHigh](#)

6.10.1 Member Typedef Documentation

6.10.1.1 _Diagnoses

```
typedef struct diagnosis::DiagnosesTable diagnosis::\_Diagnoses
```

6.10.1.2 Diagnosis_Range

```
typedef struct diagnosis::DiagnosisRange diagnosis::Diagnosis\_Range
```

6.10.2 Member Function Documentation

6.10.2.1 critCheck()

```
void diagnosis::critCheck ( ) [inline]
```

6.10.2.2 critRangeAlert()

```
virtual void diagnosis::critRangeAlert ( ) [virtual]
```

6.10.2.3 determineDiagnosis()

```
std::string diagnosis::determineDiagnosis ( ) [inline]
```

6.10.2.4 displayDiagnosis()

```
virtual void diagnosis::displayDiagnosis ( ) [virtual]
```

6.10.2.5 findMinMax()

```
void diagnosis::findMinMax ( ) [inline]
```

6.10.2.6 SetdiagnosisRanges()

```
void diagnosis::SetdiagnosisRanges (
    float minimum,
    float maximum,
    std::string diagnosis )
```

6.10.3 Member Data Documentation

6.10.3.1 CriticalHigh

```
float diagnosis::CriticalHigh
```

6.10.3.2 CriticalLow

```
float diagnosis::CriticalLow
```

6.10.3.3 CustomDiagnoses

```
\_Diagnoses diagnosis::CustomDiagnoses
```

6.10.3.4 CustomDiagnosis

```
Diagnosis\_Range diagnosis::CustomDiagnosis
```

6.10.3.5 stdDiagnoses

```
\_Diagnoses diagnosis::stdDiagnoses
```

6.10.3.6 stdDiagnosis

`Diagnosis_Range` `diagnosis::stdDiagnosis`

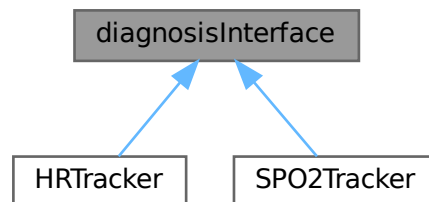
The documentation for this class was generated from the following files:

- `/home/sitcomlab/Projects/VigiSense/src/Diagnosis.h`
- `/home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp`

6.11 diagnosisInterface Class Reference

```
#include <DiagnosisInterface.h>
```

Inheritance diagram for `diagnosisInterface`:



Public Member Functions

- virtual void `start` ()=0
- virtual void `stop` ()=0
- virtual void `ping` ()=0
- virtual int `getVal` ()=0

Static Public Member Functions

- static std::string `determineSymptom` (std::vector< `symptomRange` > `symptomRanges`, int `val`)

Protected Attributes

- std::vector< `symptomRange` > `symptomRanges`

6.11.1 Member Function Documentation

6.11.1.1 determineSymptom()

```
static std::string diagnosisInterface::determineSymptom (
    std::vector< symptomRange > symptomRanges,
    int val ) [inline], [static]
```

6.11.1.2 getVal()

```
virtual int diagnosisInterface::getVal ( ) [pure virtual]
```

Implemented in [HRTracker](#), and [SPO2Tracker](#).

6.11.1.3 ping()

```
virtual void diagnosisInterface::ping ( ) [pure virtual]
```

Implemented in [HRTracker](#), and [SPO2Tracker](#).

6.11.1.4 start()

```
virtual void diagnosisInterface::start ( ) [pure virtual]
```

Implemented in [HRTracker](#), and [SPO2Tracker](#).

6.11.1.5 stop()

```
virtual void diagnosisInterface::stop ( ) [pure virtual]
```

Implemented in [HRTracker](#), and [SPO2Tracker](#).

6.11.2 Member Data Documentation

6.11.2.1 symptomRanges

```
std::vector<symptomRange> diagnosisInterface::symptomRanges [protected]
```

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

- [/home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h](#)

6.12 diagnosis::DiagnosisRange Struct Reference

```
#include <Diagnosis.h>
```

Public Attributes

- float [min](#)
- float [max](#)
- std::string [diagnosis](#)

6.12.1 Member Data Documentation

6.12.1.1 diagnosis

```
std::string diagnosis::DiagnosisRange::diagnosis
```

6.12.1.2 max

```
float diagnosis::DiagnosisRange::max
```

6.12.1.3 min

```
float diagnosis::DiagnosisRange::min
```

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

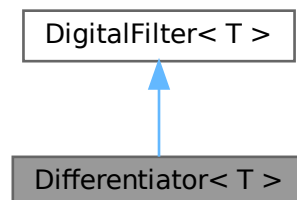
- </home/sitcomlab/Projects/VigiSense/src/Diagnosis.h>

6.13 Differentiator< T > Class Template Reference

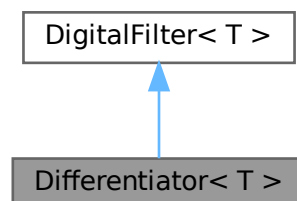
Class for differentiator.

```
#include <DigitalFilters.h>
```

Inheritance diagram for Differentiator< T >:



Collaboration diagram for Differentiator< T >:



Public Member Functions

- [Differentiator](#) (T [sampleTime](#))
- T [update](#) (T input)
- T [getOutput](#) ()

Private Attributes

- const T [sampleTime](#)
- T [y](#) = 0
- T [x1](#) = 0

6.13.1 Detailed Description

```
template<typename T>  
class Differentiator< T >
```

Class for differentiator.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 Differentiator()

```
template<typename T >  
Differentiator< T >::Differentiator (  
    T sampleTime ) [inline]
```

6.13.3 Member Function Documentation

6.13.3.1 getOutput()

```
template<typename T >  
T Differentiator< T >::getOutput ( ) [inline], [virtual]
```

Implements [DigitalFilter< T >](#).

6.13.3.2 update()

```
template<typename T >  
T Differentiator< T >::update (  
    T input ) [inline], [virtual]
```

Implements [DigitalFilter< T >](#).

6.13.4 Member Data Documentation

6.13.4.1 sampleTime

```
template<typename T >
const T Differentiator< T >::sampleTime [private]
```

6.13.4.2 x1

```
template<typename T >
T Differentiator< T >::x1 = 0 [private]
```

6.13.4.3 y

```
template<typename T >
T Differentiator< T >::y = 0 [private]
```

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

- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.14 DigitalFilter< Type > Class Template Reference

Abstract base class for digital moving filters.

```
#include <DigitalFilters.h>
```

Public Member Functions

- virtual Type [update](#) (Type newValue)=0
- virtual Type [getOutput](#) ()=0

6.14.1 Detailed Description

```
template<typename Type>
class DigitalFilter< Type >
```

Abstract base class for digital moving filters.

```
Moving filter are real time filter used for applications where
continuous filtering is necessary as it can be part of an control
system.
```

Template Parameters

<i>Type</i>	Floating point type used.
-------------	---------------------------

6.14.2 Member Function Documentation

6.14.2.1 `getOutput()`

```
template<typename Type >
virtual Type DigitalFilter< Type >::getOutput ( ) [pure virtual]
```

Implemented in [Differentiator< T >](#), [LowPassFilter](#), [LowPassFilter2](#), [HighPassFilter](#), [HighPassFilter3](#), [LowPassFilter3](#), [LowPassFilter3MatchedZ](#), and [LowPassFilter3DiffApprox](#).

6.14.2.2 `update()`

```
template<typename Type >
virtual Type DigitalFilter< Type >::update (
    Type newValue ) [pure virtual]
```

Implemented in [LowPassFilter](#), [LowPassFilter2](#), [HighPassFilter](#), [HighPassFilter3](#), [LowPassFilter3](#), [LowPassFilter3MatchedZ](#), [LowPassFilter3DiffApprox](#), and [Differentiator< T >](#).

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

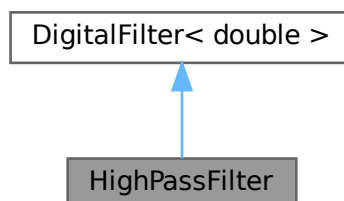
- [/home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h](#)

6.15 HighPassFilter Class Reference

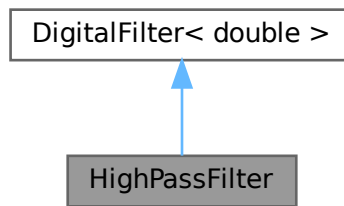
Class for high pass filter using bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for HighPassFilter:



Collaboration diagram for HighPassFilter:



Public Member Functions

- [HighPassFilter](#) (double idt, double omega_c)
Constructor to set sample time and the tau constant.
- double [update](#) (double newValue) final
Update function to push new value into the low pass filter.
- double [getOutput](#) () final
Gets the output.
- void [configOutput](#) (double newOutput)
Force the output to a desired value.
- const double * [outputPointer](#) ()

Private Attributes

- const double [amplFac](#)
- const double [y1c](#)
- const double [dt](#)
- double [x1](#) = 0
- double [output](#) = 0

6.15.1 Detailed Description

Class for high pass filter using bilinear transform.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 HighPassFilter()

```
HighPassFilter::HighPassFilter (  
    double idt,  
    double omega_c ) [inline]
```

Constructor to set sample time and the tau constant.

Parameters

in	<i>idt</i>	Sample time for the low pass filter
in	<i>itua</i> <i>_c</i>	Or τ_c The time constant for the filter. Note that $\tau_c = \frac{1}{2\pi f_c}$ where f_c is the cutoff frequency

6.15.3 Member Function Documentation

6.15.3.1 configOutput()

```
void HighPassFilter::configOutput (
    double newOutput ) [inline]
```

Force the output to a desired value.

X

This can be useful when the output needs to be forced in case of extreme inputs or such

Parameters

in	<i>newOutput</i>	The new output
----	------------------	----------------

6.15.3.2 getOutput()

```
double HighPassFilter::getOutput ( ) [inline], [final], [virtual]
```

Gets the output.

Returns

The output.

Implements [DigitalFilter< double >](#).

6.15.3.3 outputPointer()

```
const double * HighPassFilter::outputPointer ( ) [inline]
```

6.15.3.4 update()

```
double HighPassFilter::update (
    double newValue ) [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

Parameters

<code>in</code>	<code>newValue</code>	The new value after dt time
-----------------	-----------------------	-----------------------------

Returns

The new output value

Implements [DigitalFilter< double >](#).

6.15.4 Member Data Documentation

6.15.4.1 `amplFac`

```
const double HighPassFilter::amplFac [private]
```

6.15.4.2 `dt`

```
const double HighPassFilter::dt [private]
```

6.15.4.3 `output`

```
double HighPassFilter::output = 0 [private]
```

6.15.4.4 `x1`

```
double HighPassFilter::x1 = 0 [private]
```

6.15.4.5 `y1c`

```
const double HighPassFilter::y1c [private]
```

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

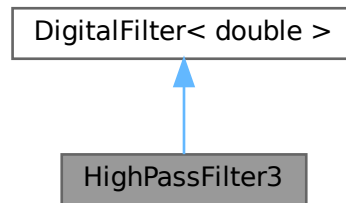
- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.16 HighPassFilter3 Class Reference

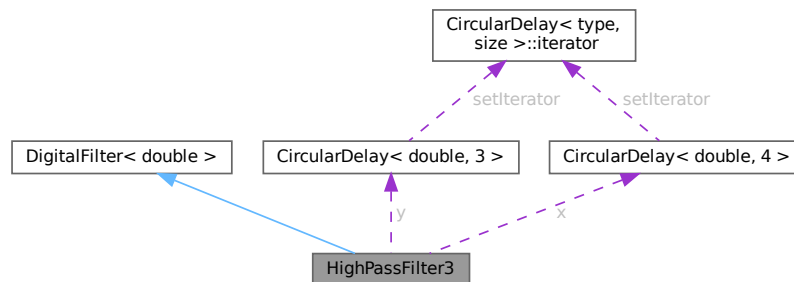
Class for third order high pass filter. This is designed using the bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for HighPassFilter3:



Collaboration diagram for HighPassFilter3:



Public Member Functions

- `HighPassFilter3` (double sampleTime, double omega_c, double ioutput=0)
- double `update` (double newValue) final
- double `getOutput` () final

Private Attributes

- const double `xc` [4]
- const double `yc` [4]
- `CircularDelay< double, 3 >` `y`
- `CircularDelay< double, 4 >` `x`

6.16.1 Detailed Description

Class for third order high pass filter. This is designed using the bilinear transform.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 HighPassFilter3()

```
HighPassFilter3::HighPassFilter3 (
    double sampleTime,
    double omega_c,
    double ioutput = 0 ) [inline]
```

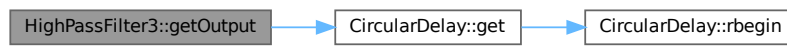
6.16.3 Member Function Documentation

6.16.3.1 getOutput()

```
double HighPassFilter3::getOutput ( ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:

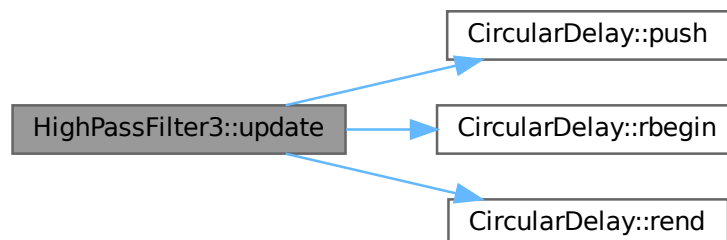


6.16.3.2 update()

```
double HighPassFilter3::update (
    double newValue ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.16.4 Member Data Documentation

6.16.4.1 x

```
CircularDelay<double, 4> HighPassFilter3::x [private]
```

6.16.4.2 xc

```
const double HighPassFilter3::xc[4] [private]
```

6.16.4.3 y

```
CircularDelay<double, 3> HighPassFilter3::y [private]
```

6.16.4.4 yc

```
const double HighPassFilter3::yc[4] [private]
```

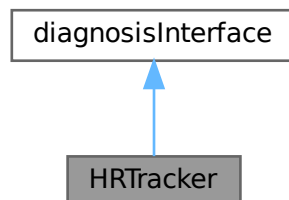
The documentation for this class was generated from the following file:

- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

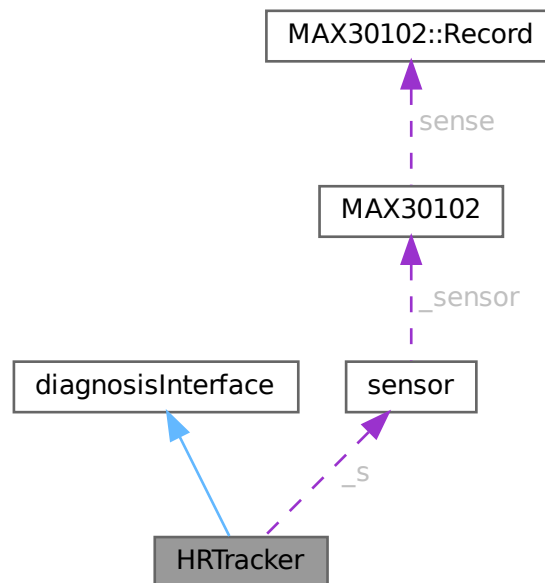
6.17 HRTracker Class Reference

```
#include <HRTracker.h>
```

Inheritance diagram for HRTracker:



Collaboration diagram for HRTracker:



Public Member Functions

- `HRTracker (sensor *s)`
- `~HRTracker ()`
- `void start ()`
- `void stop ()`
- `void ping ()`
- `int getVal ()`
- `void tracker ()`

Protected Member Functions

- `void pingThread ()`

Protected Attributes

- `sensor * _s`
- `bool threadRunning = false`
- `std::vector< symptomRange > symptomRanges`

Protected Attributes inherited from `diagnosisInterface`

- `std::vector< symptomRange > symptomRanges`

Additional Inherited Members

Static Public Member Functions inherited from [diagnosisInterface](#)

- static std::string [determineSymptom](#) (std::vector< [symptomRange](#) > [symptomRanges](#), int val)

6.17.1 Constructor & Destructor Documentation

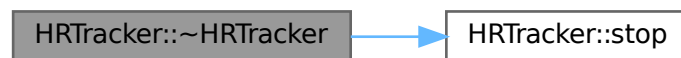
6.17.1.1 HRTracker()

```
HRTracker::HRTracker (
    sensor * s )
```

6.17.1.2 ~HRTracker()

```
HRTracker::~~HRTracker ( )
```

Here is the call graph for this function:



6.17.2 Member Function Documentation

6.17.2.1 getVal()

```
int HRTracker::getVal ( ) [virtual]
```

Implements [diagnosisInterface](#).

Here is the call graph for this function:

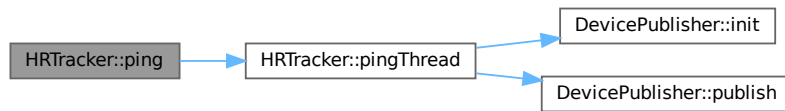


6.17.2.2 ping()

```
void HRTracker::ping ( ) [virtual]
```

Implements [diagnosisInterface](#).

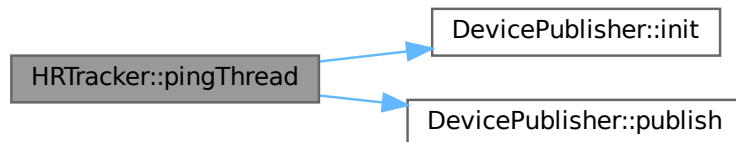
Here is the call graph for this function:



6.17.2.3 pingThread()

```
void HRTracker::pingThread ( ) [protected]
```

Here is the call graph for this function:

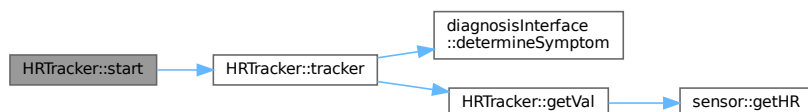


6.17.2.4 start()

```
void HRTracker::start ( ) [virtual]
```

Implements [diagnosisInterface](#).

Here is the call graph for this function:



6.17.2.5 stop()

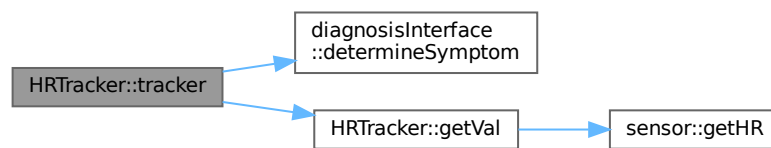
```
void HRTracker::stop ( ) [virtual]
```

Implements [diagnosisInterface](#).

6.17.2.6 tracker()

```
void HRTracker::tracker ( )
```

Here is the call graph for this function:



6.17.3 Member Data Documentation

6.17.3.1 _s

```
sensor* HRTracker::_s [protected]
```

6.17.3.2 symptomRanges

```
std::vector<symptomRange> HRTracker::symptomRanges [protected]
```

Initial value:

```
{
    {0,60,"Bradycardia"},
    {60,100,"Normal resting heart rate"},
    {100,200,"Tachycardia"}}
```

6.17.3.3 threadRunning

```
bool HRTracker::threadRunning = false [protected]
```

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

- [/home/sitcomlab/Projects/VigiSense/src/HRTracker.h](#)
- [/home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp](#)

6.18 i2c_msg Struct Reference

```
#include <i2c-dev.h>
```

Public Attributes

- `__u16` [addr](#)
- unsigned short [flags](#)
- short [len](#)
- char * [buf](#)

6.18.1 Member Data Documentation

6.18.1.1 addr

```
__u16 i2c_msg::addr
```

6.18.1.2 buf

```
char* i2c_msg::buf
```

6.18.1.3 flags

```
unsigned short i2c_msg::flags
```

6.18.1.4 len

```
short i2c_msg::len
```

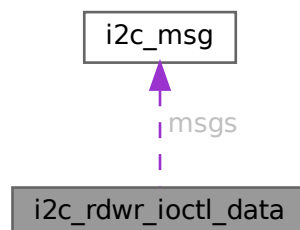
The documentation for this struct was generated from the following file:

- [/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h](#)

6.19 i2c_rdwr_ioctl_data Struct Reference

```
#include <i2c-dev.h>
```

Collaboration diagram for i2c_rdwr_ioctl_data:



Public Attributes

- struct [i2c_msg](#) * [msgs](#)
- [__u32](#) [nmsgs](#)

6.19.1 Member Data Documentation

6.19.1.1 msgs

```
struct i2c\_msg* i2c_rdwr_ioctl_data::msgs
```

6.19.1.2 nmsgs

```
\_\_u32 i2c_rdwr_ioctl_data::nmsgs
```

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

- [/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h](#)

6.20 i2c_smbus_data Union Reference

```
#include <i2c-dev.h>
```

Public Attributes

- [__u8](#) [byte](#)
- [__u16](#) [word](#)
- [__u8](#) [block](#) [[I2C_SMBUS_BLOCK_MAX](#)+2]

6.20.1 Member Data Documentation

6.20.1.1 block

```
\_\_u8 i2c_smbus_data::block [I2C\_SMBUS\_BLOCK\_MAX+2]
```

6.20.1.2 byte

```
\_\_u8 i2c_smbus_data::byte
```

6.20.1.3 word

```
\_\_u16 i2c_smbus_data::word
```

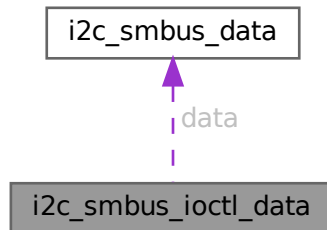
The documentation for this union was generated from the following file:

- [/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h](#)

6.21 i2c_smbus_ioctl_data Struct Reference

```
#include <i2c-dev.h>
```

Collaboration diagram for i2c_smbus_ioctl_data:



Public Attributes

- `__u8 read_write`
- `__u8 command`
- `__u32 size`
- `union i2c_smbus_data * data`

6.21.1 Member Data Documentation

6.21.1.1 command

```
__u8 i2c_smbus_ioctl_data::command
```

6.21.1.2 data

```
union i2c_smbus_data* i2c_smbus_ioctl_data::data
```

6.21.1.3 read_write

```
__u8 i2c_smbus_ioctl_data::read_write
```

6.21.1.4 size

```
__u32 i2c_smbus_ioctl_data::size
```

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

- `/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h`

6.22 CircularDelay< type, size >::iterator Class Reference

```
#include <CircularDelay.hpp>
```

Public Types

- typedef [iterator](#) [self_type](#)
- typedef std::bidirectional_iterator_tag [iterator_category](#)
- typedef int [difference_type](#)

Public Member Functions

- [iterator](#) (const [CircularDelay](#)< type, size >::iterator &it)
- [self_type](#) operator++ ()
- [self_type](#) operator++ (int)
- [self_type](#) operator-- ()
- [self_type](#) operator-- (int)
- type & [operator*](#) ()
- type * [operator->](#) ()
- type & [operator\[\]](#) (unsigned int index)
- bool [operator==](#) (const [self_type](#) &rhs)
- bool [operator!=](#) (const [self_type](#) &rhs)

Private Member Functions

- [iterator](#) (type *[data](#), type *ptr)

Private Attributes

- type * [data_](#) = nullptr
- type * [ptr_](#) = nullptr

Friends

- class [CircularDelay](#)

6.22.1 Member Typedef Documentation

6.22.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::iterator::difference_type
```

6.22.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::iterator::iterator_↔
category
```

6.22.1.3 self_type

```
template<typename type , size_t size>
typedef iterator CircularDelay< type, size >::iterator::self_type
```

6.22.2 Constructor & Destructor Documentation

6.22.2.1 iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::iterator::iterator (
    const CircularDelay< type, size >::iterator & it ) [inline]
```

6.22.2.2 iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::iterator::iterator (
    type * data,
    type * ptr ) [inline], [private]
```

6.22.3 Member Function Documentation

6.22.3.1 operator!=(())

```
template<typename type , size_t size>
bool CircularDelay< type, size >::iterator::operator!= (
    const self_type & rhs ) [inline]
```

6.22.3.2 operator*()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::iterator::operator* ( ) [inline]
```

6.22.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator++ ( ) [inline]
```

6.22.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator++ (
    int ) [inline]
```

6.22.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator-- ( ) [inline]
```

6.22.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::iterator::operator-- (
    int ) [inline]
```

6.22.3.7 operator->()

```
template<typename type , size_t size>
type * CircularDelay< type, size >::iterator::operator-> ( ) [inline]
```

6.22.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::iterator::operator== (
    const self_type & rhs ) [inline]
```

6.22.3.9 operator[]()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::iterator::operator[] (
    unsigned int index ) [inline]
```

6.22.4 Friends And Related Symbol Documentation**6.22.4.1 CircularDelay**

```
template<typename type , size_t size>
friend class CircularDelay [friend]
```

6.22.5 Member Data Documentation**6.22.5.1 data_**

```
template<typename type , size_t size>
type* CircularDelay< type, size >::iterator::data_ = nullptr [private]
```

6.22.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::iterator::ptr_ = nullptr [private]
```

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

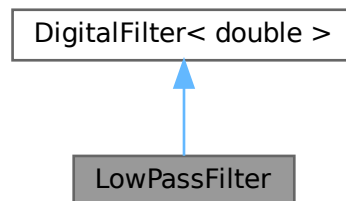
- [/home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp](#)

6.23 LowPassFilter Class Reference

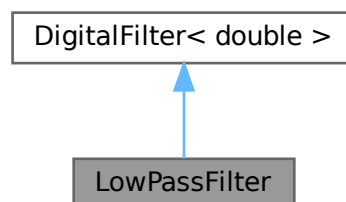
Class for a low pass filter.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter:



Collaboration diagram for LowPassFilter:



Public Member Functions

- [LowPassFilter](#) (double idt, double omega_c, double ioutput=0)
Constructor to set sample time and the tau constant.
- double [update](#) (double newValue) final
Update function to push new value into the low pass filter.
- double [getOutput](#) () final
Gets the output.
- void [configOutput](#) (double newOutput)
Force the output to a desired value.
- const double * [outputPointer](#) ()

Private Attributes

- const double [epow](#)
- const double [dt](#)
one time calculation constant
- double [output](#)

6.23.1 Detailed Description

Class for a low pass filter.

Design to be a first order Butterworth low pass filter.
Transformation done using the matched-Z-transform method

6.23.2 Constructor & Destructor Documentation

6.23.2.1 LowPassFilter()

```
LowPassFilter::LowPassFilter (
    double idt,
    double omega_c,
    double ioutput = 0 ) [inline]
```

Constructor to set sample time and the tau constant.

Parameters

in	<i>idt</i>	Sample time for the low pass filter
in	<i>itua_c</i>	Or τ_c The time constant for the filter. Note that $\tau_c = \frac{1}{2\pi f_c}$ where f_c is the cutoff frequency

6.23.3 Member Function Documentation

6.23.3.1 configOutput()

```
void LowPassFilter::configOutput (
    double newOutput ) [inline]
```

Force the output to a desired value.

This can be useful when the output needs to be forced in case of extreme inputs or such

Parameters

in	<i>newOutput</i>	The new output
----	------------------	----------------

6.23.3.2 getOutput()

```
double LowPassFilter::getOutput ( ) [inline], [final], [virtual]
```

Gets the output.

Returns

The output.

Implements [DigitalFilter< double >](#).

6.23.3.3 outputPointer()

```
const double * LowPassFilter::outputPointer ( ) [inline]
```

6.23.3.4 update()

```
double LowPassFilter::update (
    double newValue ) [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

Parameters

in	<i>newValue</i>	The new value after dt time
----	-----------------	-----------------------------

Returns

The new output value

Implements [DigitalFilter< double >](#).

6.23.4 Member Data Documentation

6.23.4.1 dt

```
const double LowPassFilter::dt [private]
```

one time calculation constant

6.23.4.2 epow

```
const double LowPassFilter::epow [private]
```

6.23.4.3 output

```
double LowPassFilter::output [private]
```

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

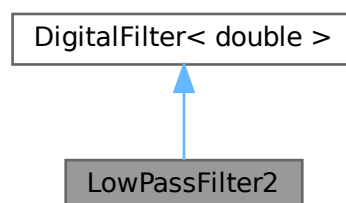
- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.24 LowPassFilter2 Class Reference

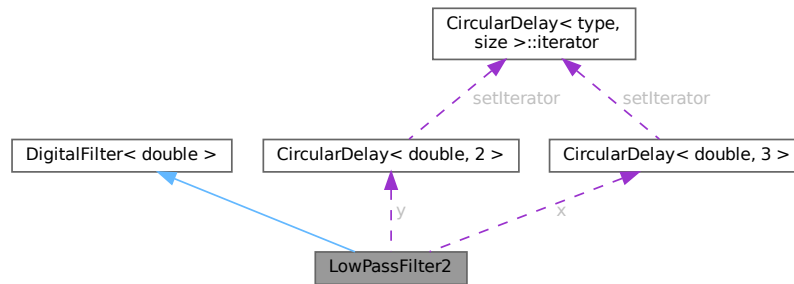
Class for a 2nd order low pass filter.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter2:



Collaboration diagram for LowPassFilter2:



Public Member Functions

- [LowPassFilter2](#) (double dt, double tau_c, double ioutput=0)
Constructor to set sample time and the tau constant.
- double [update](#) (double newValue) final
Update function to push new value into the low pass filter.
- double [getOutput](#) () final
Gets the output.
- void [configOutput](#) (double newOutput)
Force the output to a desired value.

Private Attributes

- const double [yc](#) [2]
- const double [xc](#) [3]
- [CircularDelay](#)< double, 2 > [y](#)
- [CircularDelay](#)< double, 3 > [x](#)

6.24.1 Detailed Description

Class for a 2nd order low pass filter.

Design to be a 2nd order Butterworth low pass filter.
Transformation done using the bilinear transform method

6.24.2 Constructor & Destructor Documentation

6.24.2.1 LowPassFilter2()

```

LowPassFilter2::LowPassFilter2 (
    double dt,
    double tau_c,
    double ioutput = 0 ) [inline]
  
```

Constructor to set sample time and the tau constant.

Parameters

in	<i>idt</i>	Sample time for the low pass filter
in	<i>itua</i> <i>_c</i>	Or τ_c The time constant for the filter. Note that $\tau_c = \frac{1}{2\pi f_c}$ where f_c is the cutoff frequency

6.24.3 Member Function Documentation

6.24.3.1 configOutput()

```
void LowPassFilter2::configOutput (
    double newOutput ) [inline]
```

Force the output to a desired value.

This can be useful when the output needs to be forced in case of extreme inputs or such

Parameters

in	<i>newOutput</i>	The new output
----	------------------	----------------

6.24.3.2 getOutput()

```
double LowPassFilter2::getOutput ( ) [inline], [final], [virtual]
```

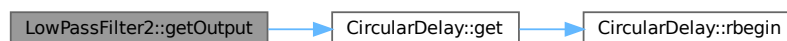
Gets the output.

Returns

The output.

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.24.3.3 update()

```
double LowPassFilter2::update (
    double newValue ) [inline], [final], [virtual]
```

Update function to push new value into the low pass filter.

Parameters

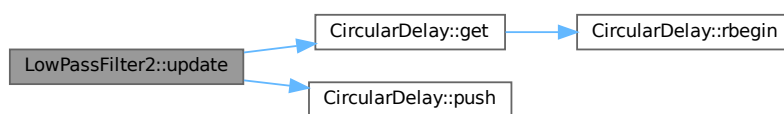
in	<i>newValue</i>	The new value after dt time
----	-----------------	-----------------------------

Returns

The new output value

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.24.4 Member Data Documentation

6.24.4.1 x

```
CircularDelay<double, 3> LowPassFilter2::x [private]
```

6.24.4.2 xc

```
const double LowPassFilter2::xc[3] [private]
```

6.24.4.3 y

```
CircularDelay<double, 2> LowPassFilter2::y [private]
```

6.24.4.4 yc

```
const double LowPassFilter2::yc[2] [private]
```

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

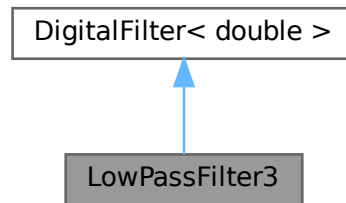
- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.25 LowPassFilter3 Class Reference

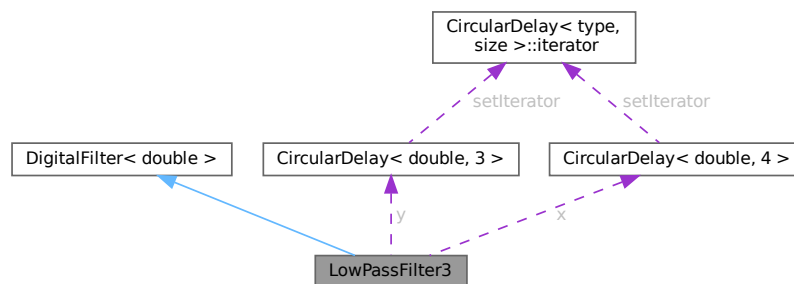
Class for third order high pass filter. This is designed using the bilinear transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3:



Collaboration diagram for LowPassFilter3:



Public Member Functions

- [LowPassFilter3](#) (long double sampleTime, long double omega_c, long double ioutput=0)
- double [update](#) (double newValue) final
- double [getOutput](#) () final

Private Attributes

- const double [yc](#) [4]
- const double [xc](#) [4]
- [CircularDelay](#)< double, 3 > [y](#)
- [CircularDelay](#)< double, 4 > [x](#)

6.25.1 Detailed Description

Class for third order high pass filter. This is designed using the bilinear transform.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 LowPassFilter3()

```
LowPassFilter3::LowPassFilter3 (  
    long double sampleTime,  
    long double omega_c,  
    long double ioutput = 0 ) [inline]
```

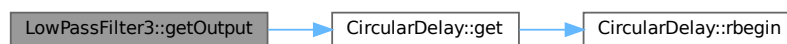
6.25.3 Member Function Documentation

6.25.3.1 getOutput()

```
double LowPassFilter3::getOutput ( ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:

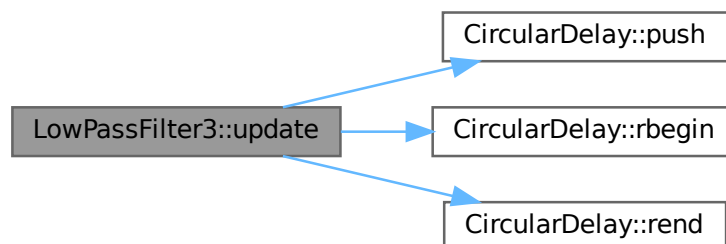


6.25.3.2 update()

```
double LowPassFilter3::update (  
    double newValue ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.25.4 Member Data Documentation

6.25.4.1 x

```
CircularDelay<double, 4> LowPassFilter3::x [private]
```

6.25.4.2 xc

```
const double LowPassFilter3::xc[4] [private]
```

6.25.4.3 y

```
CircularDelay<double, 3> LowPassFilter3::y [private]
```

6.25.4.4 yc

```
const double LowPassFilter3::yc[4] [private]
```

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

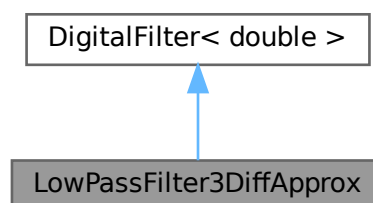
- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.26 LowPassFilter3DiffApprox Class Reference

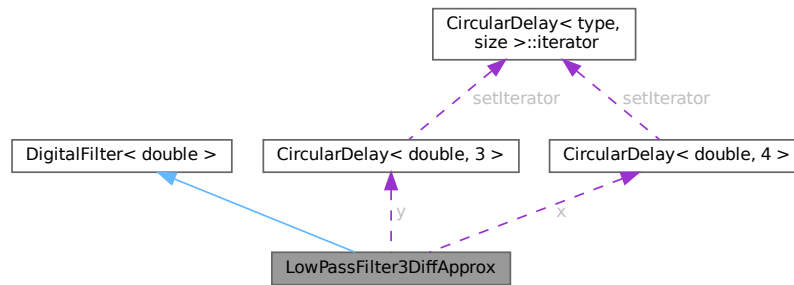
Class for third order high pass filter. This is designed using the approximated differtial approach where $s=(Z-1)/(Z*T)$.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3DiffApprox:



Collaboration diagram for LowPassFilter3DiffApprox:



Public Member Functions

- [LowPassFilter3DiffApprox](#) (double sampleTime, double omega_c, double ioutput=0)
- double [update](#) (double newValue) final
- double [getOutput](#) () final

Private Attributes

- const double [xc](#) [4]
- const double [yc](#) [4]
- [CircularDelay](#)< double, 3 > [y](#)
- [CircularDelay](#)< double, 4 > [x](#)

6.26.1 Detailed Description

Class for third order high pass filter. This is designed using the approximated differtial approach where $s=(Z-1)/(Z*T)$.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 LowPassFilter3DiffApprox()

```

LowPassFilter3DiffApprox::LowPassFilter3DiffApprox (
    double sampleTime,
    double omega_c,
    double ioutput = 0 ) [inline]
  
```

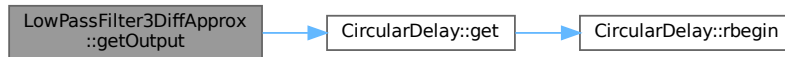
6.26.3 Member Function Documentation

6.26.3.1 `getOutput()`

```
double LowPassFilter3DiffApprox::getOutput ( ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:

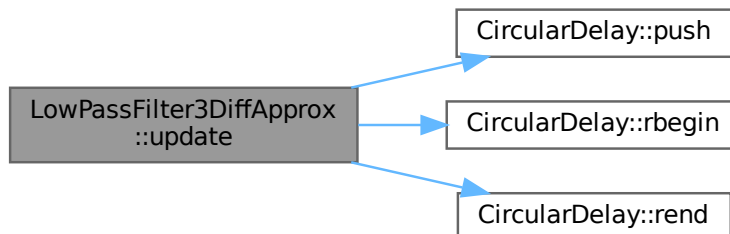


6.26.3.2 `update()`

```
double LowPassFilter3DiffApprox::update (
    double newValue ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.26.4 Member Data Documentation

6.26.4.1 `x`

```
CircularDelay<double, 4> LowPassFilter3DiffApprox::x [private]
```

6.26.4.2 `xc`

```
const double LowPassFilter3DiffApprox::xc[4] [private]
```


6.26.4.3 y

```
CircularDelay<double, 3> LowPassFilter3DiffApprox::y [private]
```

6.26.4.4 yc

```
const double LowPassFilter3DiffApprox::yc[4] [private]
```

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

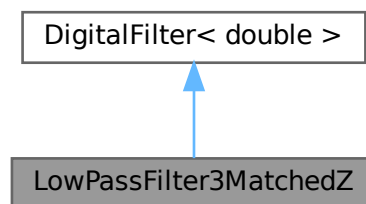
- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.27 LowPassFilter3MatchedZ Class Reference

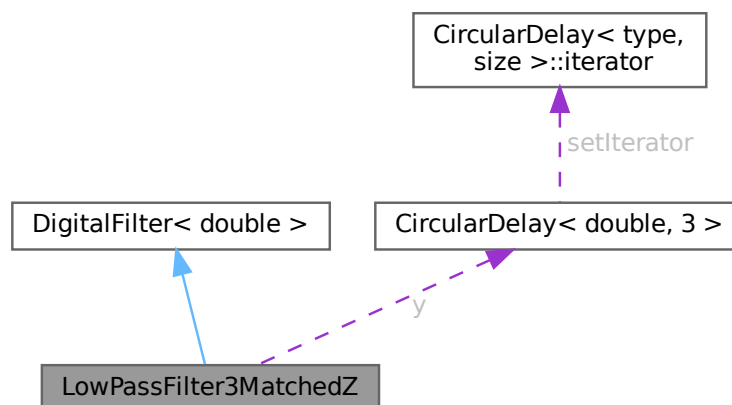
Class for third order high pass filter. This is designed using the matched Z transform.

```
#include <DigitalFilters.h>
```

Inheritance diagram for LowPassFilter3MatchedZ:



Collaboration diagram for LowPassFilter3MatchedZ:



Public Member Functions

- [LowPassFilter3MatchedZ](#) (long double sampleTime, long double omega_c)
- double [update](#) (double newValue) final
- double [getOutput](#) () final

Private Attributes

- const double [amplFac](#)
- const double [yc](#) [3]
- [CircularDelay](#)< double, 3 > [y](#)

6.27.1 Detailed Description

Class for third order high pass filter. This is designed using the matched Z transform.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 LowPassFilter3MatchedZ()

```
LowPassFilter3MatchedZ::LowPassFilter3MatchedZ (
    long double sampleTime,
    long double omega_c ) [inline]
```

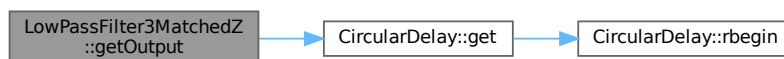
6.27.3 Member Function Documentation

6.27.3.1 getOutput()

```
double LowPassFilter3MatchedZ::getOutput ( ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:

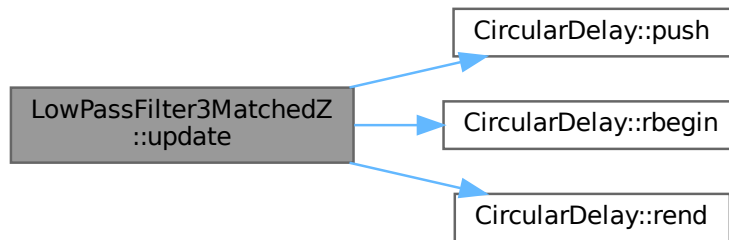


6.27.3.2 update()

```
double LowPassFilter3MatchedZ::update (
    double newValue ) [inline], [final], [virtual]
```

Implements [DigitalFilter< double >](#).

Here is the call graph for this function:



6.27.4 Member Data Documentation

6.27.4.1 amplFac

```
const double LowPassFilter3MatchedZ::amplFac [private]
```

6.27.4.2 y

```
CircularDelay<double, 3> LowPassFilter3MatchedZ::y [private]
```

6.27.4.3 yc

```
const double LowPassFilter3MatchedZ::yc[3] [private]
```

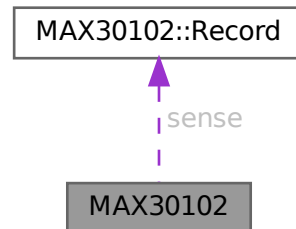
The documentation for this class was generated from the following file:

- </home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h>

6.28 MAX30102 Class Reference

```
#include <MAX30102.h>
```

Collaboration diagram for MAX30102:



Classes

- struct [Record](#)

Public Member Functions

- [MAX30102](#) (void)
- int [begin](#) (uint32_t i2cSpeed=[I2C_SPEED_STANDARD](#), uint8_t i2cAddr=[MAX30102_ADDRESS](#))
- uint32_t [getRed](#) (void)
- uint32_t [getIR](#) (void)
- bool [safeCheck](#) (uint8_t maxTimeToCheck)
- void [wakeUp](#) ()
- void [shutDown](#) ()
- void [softReset](#) ()
- void [setLEDMode](#) (uint8_t mode)
- void [setADCRange](#) (uint8_t adcRange)
- void [setSampleRate](#) (uint8_t sampleRate)
- void [setPulseWidth](#) (uint8_t pulseWidth)
- void [setPulseAmplitudeRed](#) (uint8_t value)
- void [setPulseAmplitudeIR](#) (uint8_t value)
- void [setPulseAmplitudeProximity](#) (uint8_t value)
- void [setProximityThreshold](#) (uint8_t thresMSB)
- void [enableSlot](#) (uint8_t slotNumber, uint8_t device)
- void [disableSlots](#) (void)
- uint8_t [getINT1](#) (void)
- uint8_t [getINT2](#) (void)
- void [enableAFULL](#) (void)
- void [disableAFULL](#) (void)
- void [enableDATARDY](#) (void)
- void [disableDATARDY](#) (void)
- void [enableALCOVF](#) (void)
- void [disableALCOVF](#) (void)

- void [enablePROXINT](#) (void)
- void [disablePROXINT](#) (void)
- void [enableDIETEMPRDY](#) (void)
- void [disableDIETEMPRDY](#) (void)
- void [setFIFOAverage](#) (uint8_t samples)
- void [enableFIFORollover](#) ()
- void [disableFIFORollover](#) ()
- void [setFIFOAlmostFull](#) (uint8_t samples)
- uint16_t [check](#) (void)
- uint8_t [available](#) (void)
- void [nextSample](#) (void)
- uint32_t [getFIFORed](#) (void)
- uint32_t [getFIFOIR](#) (void)
- uint8_t [getWritePointer](#) (void)
- uint8_t [getReadPointer](#) (void)
- void [clearFIFO](#) (void)
- void [setPROXINTTHRESH](#) (uint8_t val)
- float [readTemperature](#) ()
- float [readTemperatureF](#) ()
- uint8_t [getRevisionID](#) ()
- uint8_t [readPartID](#) ()
- virtual void [hasSample](#) ()
- void [setup](#) (uint8_t powerLevel=0x1F, uint8_t sampleAverage=4, uint8_t ledMode=2, int sampleRate=400, int pulseWidth=411, int adcRange=4096)

Private Types

- typedef struct [MAX30102::Record sense_struct](#)

Private Member Functions

- void [readRevisionID](#) ()
- void [bitMask](#) (uint8_t reg, uint8_t mask, uint8_t thing)
- std::vector< uint8_t > [readMany](#) (uint8_t address, uint8_t length)
- void [dataReady](#) ()

Static Private Member Functions

- static void [gpioISR](#) (int, int, uint32_t, void *userdata)

Private Attributes

- int [_i2c](#)
- uint8_t [_i2caddr](#)
- uint8_t [activeLEDs](#)
- uint8_t [revisionID](#)
- [sense_struct](#) [sense](#)

6.28.1 Member Typedef Documentation

6.28.1.1 sense_struct

```
typedef struct MAX30102::Record MAX30102::sense_struct [private]
```

6.28.2 Constructor & Destructor Documentation

6.28.2.1 MAX30102()

```
MAX30102::MAX30102 (  
    void )
```

6.28.3 Member Function Documentation

6.28.3.1 available()

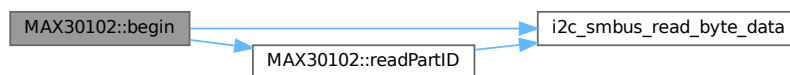
```
uint8_t MAX30102::available (  
    void )
```

Returns the number of samples available.

6.28.3.2 begin()

```
int MAX30102::begin (  
    uint32_t i2cSpeed = I2C_SPEED_STANDARD,  
    uint8_t i2cAddr = MAX30102_ADDRESS )
```

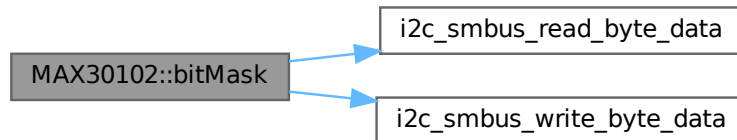
Initializes sensor. Returns negative number on failure. Returns sensor revision on success. Here is the call graph for this function:



6.28.3.3 bitMask()

```
void MAX30102::bitMask (
    uint8_t reg,
    uint8_t mask,
    uint8_t thing ) [private]
```

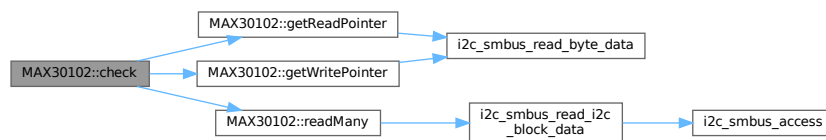
Set certain thing in register. Here is the call graph for this function:



6.28.3.4 check()

```
uint16_t MAX30102::check (
    void )
```

Here is the call graph for this function:



6.28.3.5 clearFIFO()

```
void MAX30102::clearFIFO (
    void )
```

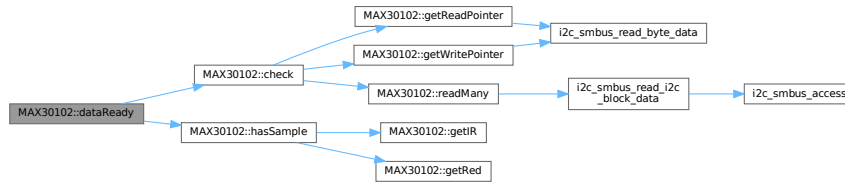
Resets all points to start in a known state. Recommended to clear FIFO before beginning a read. Here is the call graph for this function:



6.28.3.6 dataReady()

```
void MAX30102::dataReady ( ) [private]
```

Here is the call graph for this function:



6.28.3.7 disableAFULL()

```
void MAX30102::disableAFULL (
    void )
```

Here is the call graph for this function:



6.28.3.8 disableALCOVF()

```
void MAX30102::disableALCOVF (
    void )
```

Here is the call graph for this function:



6.28.3.9 disableDATARDY()

```
void MAX30102::disableDATARDY (
    void )
```

Here is the call graph for this function:



6.28.3.10 disableDIETEMPRDY()

```
void MAX30102::disableDIETEMPRDY (
    void )
```

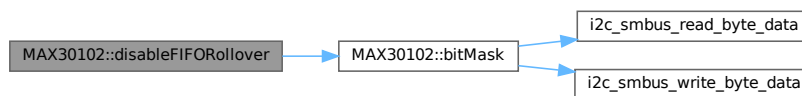
Here is the call graph for this function:



6.28.3.11 disableFIFORollover()

```
void MAX30102::disableFIFORollover (
    void )
```

Disable roll over if FIFO over flows. Here is the call graph for this function:



6.28.3.12 disablePROXINT()

```
void MAX30102::disablePROXINT (  
    void )
```

Here is the call graph for this function:



6.28.3.13 disableSlots()

```
void MAX30102::disableSlots (  
    void )
```

Clears all slot assignments. Here is the call graph for this function:



6.28.3.14 enableAFULL()

```
void MAX30102::enableAFULL (  
    void )
```

Here is the call graph for this function:



6.28.3.15 enableALCOVF()

```
void MAX30102::enableALCOVF (  
    void )
```

Here is the call graph for this function:



6.28.3.16 enableDATARDY()

```
void MAX30102::enableDATARDY (  
    void )
```

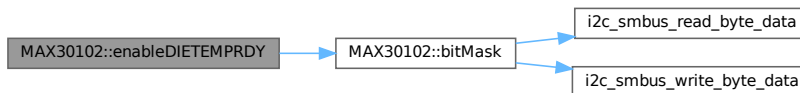
Here is the call graph for this function:



6.28.3.17 enableDIETEMPRDY()

```
void MAX30102::enableDIETEMPRDY (  
    void )
```

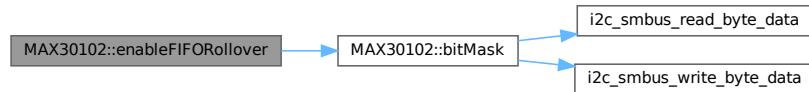
Here is the call graph for this function:



6.28.3.18 enableFIFORollover()

```
void MAX30102::enableFIFORollover (
    void )
```

Enable roll over if FIFO over flows. Here is the call graph for this function:



6.28.3.19 enablePROXINT()

```
void MAX30102::enablePROXINT (
    void )
```

Here is the call graph for this function:



6.28.3.20 enableSlot()

```
void MAX30102::enableSlot (
    uint8_t slotNumber,
    uint8_t device )
```

Given a slot number assign a thing to it. Devices are `SLOT_RED_LED` or `SLOT_RED_PILOT` (proximity) Assigning a `SLOT_RED_LED` will pulse LED Assigning a `SLOT_RED_PILOT` will ?? Here is the call graph for this function:



6.28.3.21 getFIFOIR()

```
uint32_t MAX30102::getFIFOIR (  
    void )
```

Report the next IR value in FIFO.

6.28.3.22 getFIFORed()

```
uint32_t MAX30102::getFIFORed (  
    void )
```

Report the next Red value in FIFO.

6.28.3.23 getINT1()

```
uint8_t MAX30102::getINT1 (  
    void )
```

Here is the call graph for this function:



6.28.3.24 getINT2()

```
uint8_t MAX30102::getINT2 (  
    void )
```

Here is the call graph for this function:



6.28.3.25 getIR()

```
uint32_t MAX30102::getIR (
    void )
```

Report the most recent IR value. change to only return without calling checks

6.28.3.26 getReadPointer()

```
uint8_t MAX30102::getReadPointer (
    void )
```

Read the FIFO Read Pointer. Here is the call graph for this function:



6.28.3.27 getRed()

```
uint32_t MAX30102::getRed (
    void )
```

Report the most recent Red value. change to only return without calling checks

6.28.3.28 getRevisionID()

```
uint8_t MAX30102::getRevisionID ( )
```

6.28.3.29 getWritePointer()

```
uint8_t MAX30102::getWritePointer (
    void )
```

Read the FIFO Write Pointer. Here is the call graph for this function:



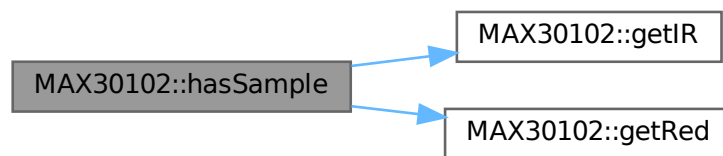
6.28.3.30 gpioISR()

```
static void MAX30102::gpioISR (
    int ,
    int ,
    uint32_t ,
    void * userdata ) [inline], [static], [private]
```

6.28.3.31 hasSample()

```
void MAX30102::hasSample ( ) [virtual]
```

Here is the call graph for this function:



6.28.3.32 nextSample()

```
void MAX30102::nextSample (
    void )
```

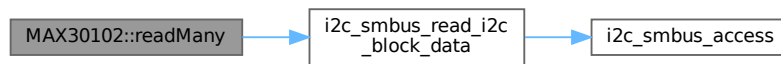
Advance the tail. Here is the call graph for this function:



6.28.3.33 readMany()

```
std::vector< uint8_t > MAX30102::readMany (
    uint8_t address,
    uint8_t length ) [private]
```

Read multiple bytes from register. Here is the call graph for this function:



6.28.3.34 readPartID()

```
uint8_t MAX30102::readPartID ( )
```

Here is the call graph for this function:



6.28.3.35 readRevisionID()

```
void MAX30102::readRevisionID ( ) [private]
```

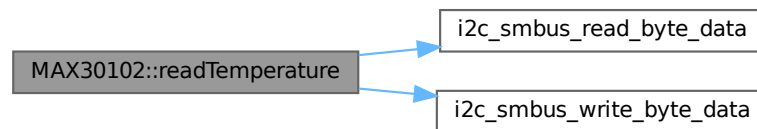
Here is the call graph for this function:



6.28.3.36 readTemperature()

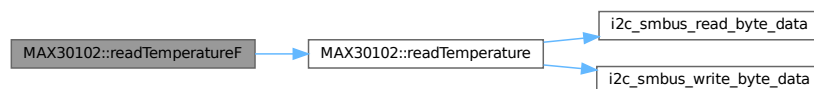
```
float MAX30102::readTemperature ( )
```

Die Temperature. Returns temperature in C. Here is the call graph for this function:

**6.28.3.37 readTemperatureF()**

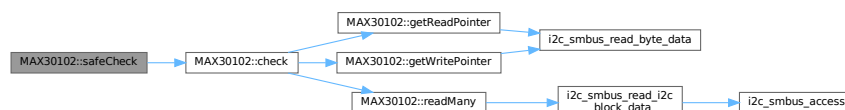
```
float MAX30102::readTemperatureF ( )
```

Returns die temperature in F. Here is the call graph for this function:

**6.28.3.38 safeCheck()**

```
bool MAX30102::safeCheck (
    uint8_t maxTimeToCheck )
```

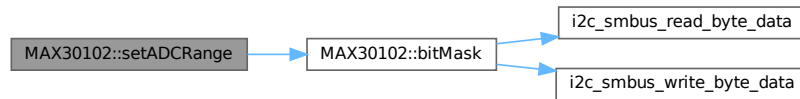
Check for new data but give up after a certain amount of time. Returns true if new data was found. Returns false if new data was not found. Here is the call graph for this function:



6.28.3.39 setADCRange()

```
void MAX30102::setADCRange (
    uint8_t adcRange )
```

Sets ADC Range. Available ADC Range: 2048, 4096, 8192, 16384 Here is the call graph for this function:



6.28.3.40 setFIFOAlmostFull()

```
void MAX30102::setFIFOAlmostFull (
    uint8_t numberOfSamples )
```

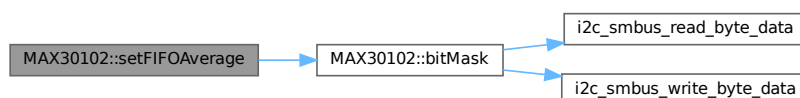
Sets number of samples to trigger the almost full interrupt. Power on default is 32 samples. Here is the call graph for this function:



6.28.3.41 setFIFOAverage()

```
void MAX30102::setFIFOAverage (
    uint8_t numberOfSamples )
```

Sets sample average. Here is the call graph for this function:



6.28.3.42 setLEDMode()

```
void MAX30102::setLEDMode (
    uint8_t mode )
```

Sets which LEDs are used for sampling.

- Red only
- Red+IR only
- Custom

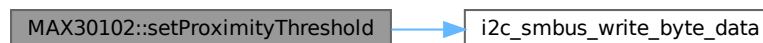
Here is the call graph for this function:



6.28.3.43 setProximityThreshold()

```
void MAX30102::setProximityThreshold (
    uint8_t threshMSB )
```

Set the IR ADC count that will trigger the beginning of particle-sensing mode. The `threshMSB` signifies only the 8 most significant-bits of the ADC count. Here is the call graph for this function:



6.28.3.44 setPROXINTTHRESH()

```
void MAX30102::setPROXINTTHRESH (
    uint8_t val )
```

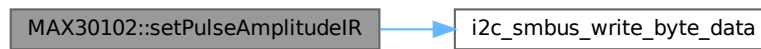
Sets the `PROX_INT_THRESHOLD`. Here is the call graph for this function:



6.28.3.45 setPulseAmplitudeIR()

```
void MAX30102::setPulseAmplitudeIR (
    uint8_t amplitude )
```

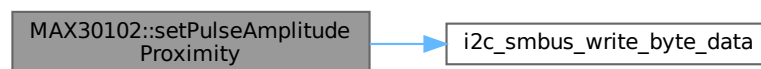
Sets IR LED Pulse Amplitude. Here is the call graph for this function:



6.28.3.46 setPulseAmplitudeProximity()

```
void MAX30102::setPulseAmplitudeProximity (
    uint8_t value )
```

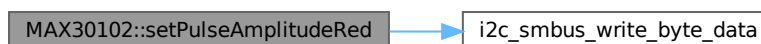
Here is the call graph for this function:



6.28.3.47 setPulseAmplitudeRed()

```
void MAX30102::setPulseAmplitudeRed (
    uint8_t amplitude )
```

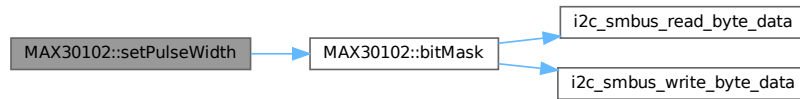
Sets Red LED Pulse Amplitude. Here is the call graph for this function:



6.28.3.48 setPulseWidth()

```
void MAX30102::setPulseWidth (
    uint8_t pulseWidth )
```

Sets Pulse Width. Available Pulse Width: 69, 188, 215, 411 Here is the call graph for this function:

**6.28.3.49 setSampleRate()**

```
void MAX30102::setSampleRate (
    uint8_t sampleRate )
```

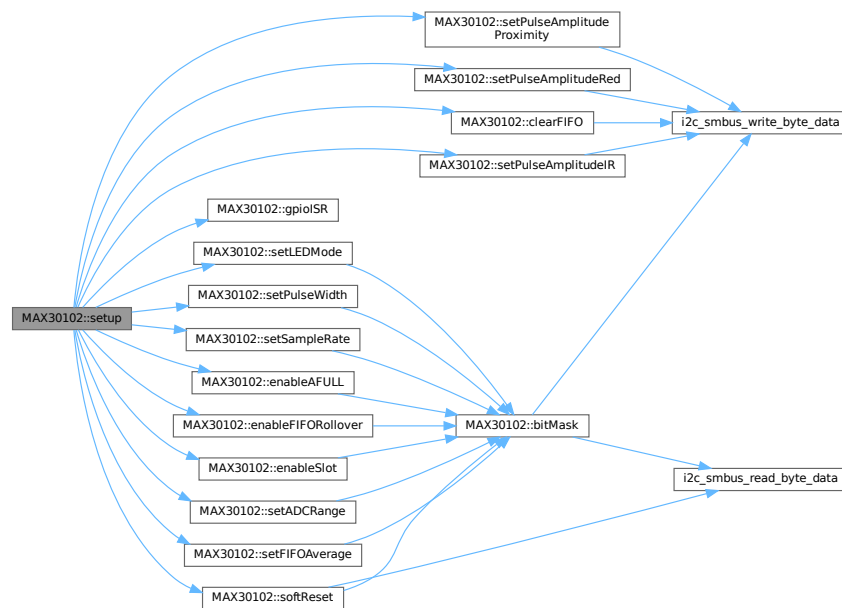
Sets Sample Rate. Available Sample Rates: 50, 100, 200, 400, 800, 1000, 1600, 3200 Here is the call graph for this function:

**6.28.3.50 setup()**

```
void MAX30102::setup (
    uint8_t powerLevel = 0x1F,
    uint8_t sampleAverage = 4,
    uint8_t ledMode = 2,
    int sampleRate = 400,
    int pulseWidth = 411,
    int adcRange = 4096 )
```

Setup the Sensor use led mode 2 where ir and red is used. Default averages 4 samples, with samepleRate of 400, resulting in effective sampleRate of 100, giving a period of 10ms between each averaged sample. Meaning that the 32 sample FIFO will fill up in approx 320ms

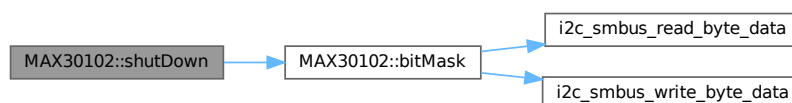
Defaults: powerLevel = 0x1F (6.2mA) sampleAverage = 4 ledMode = 2 sampleRate = 400 pulseWidth = 411 adc↔
Range = 4096 Here is the call graph for this function:



6.28.3.51 shutDown()

```
void MAX30102::shutDown (
    void )
```

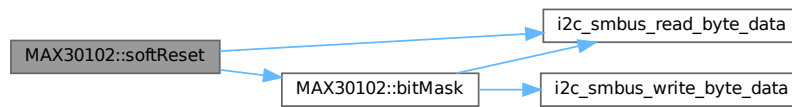
Put sensor into low power mode. During this mode the sensor will continue to respond to I2C commands but will not update or take new readings, such as temperature. Here is the call graph for this function:



6.28.3.52 softReset()

```
void MAX30102::softReset (
    void )
```

All configuration, threshold, and data registers are reset to their power-on state through a power-on reset. The reset bit is cleared back to zero after reset finishes. Here is the call graph for this function:



6.28.3.53 `wakeUp()`

```
void MAX30102::wakeUp (
    void )
```

Pull sensor out of low power mode. Here is the call graph for this function:



6.28.4 Member Data Documentation

6.28.4.1 `_i2c`

```
int MAX30102::_i2c [private]
```

6.28.4.2 `_i2caddr`

```
uint8_t MAX30102::_i2caddr [private]
```

6.28.4.3 `activeLEDs`

```
uint8_t MAX30102::activeLEDs [private]
```

6.28.4.4 `revisionID`

```
uint8_t MAX30102::revisionID [private]
```

6.28.4.5 `sense`

```
sense_struct MAX30102::sense [private]
```

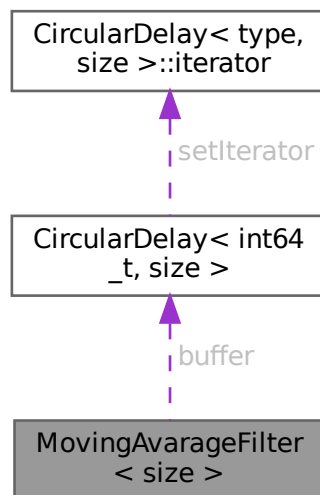
The documentation for this class was generated from the following files:

- `/home/sitcomlab/Projects/VigiSense/src/MAX30102.h`
- `/home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp`

6.29 `MovingAvarageFilter< size >` Class Template Reference

```
#include <DigitalFilters.h>
```

Collaboration diagram for `MovingAvarageFilter< size >`:



Public Member Functions

- double `update` (double input)

Private Attributes

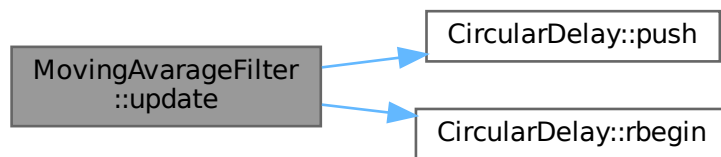
- `int64_t output` = 0
- `CircularDelay< int64_t, size > buffer`

6.29.1 Member Function Documentation

6.29.1.1 update()

```
template<size_t size>
double MovingAvarageFilter< size >::update (
    double input ) [inline]
```

Here is the call graph for this function:



6.29.2 Member Data Documentation

6.29.2.1 buffer

```
template<size_t size>
CircularDelay<int64_t, size> MovingAvarageFilter< size >::buffer [private]
```

6.29.2.2 output

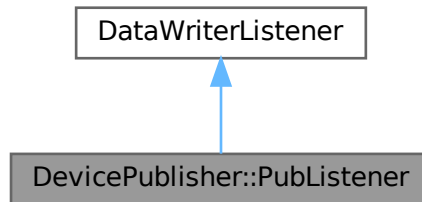
```
template<size_t size>
int64_t MovingAvarageFilter< size >::output = 0 [private]
```

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

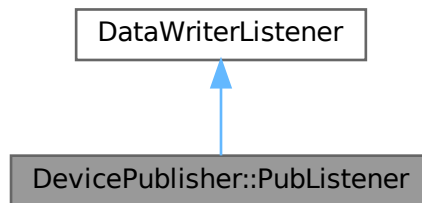
- [/home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h](#)

6.30 DevicePublisher::PubListener Class Reference

Inheritance diagram for DevicePublisher::PubListener:



Collaboration diagram for DevicePublisher::PubListener:



Public Member Functions

- [PubListener](#) ()
- [~PubListener](#) () override
- void [on_publication_matched](#) (DataWriter *, const PublicationMatchedStatus &info) override

Public Attributes

- std::atomic_int [matched_](#)

6.30.1 Constructor & Destructor Documentation

6.30.1.1 PubListener()

```
DevicePublisher::PubListener::PubListener ( ) [inline]
```

6.30.1.2 ~PubListener()

```
DevicePublisher::PubListener::~~PubListener ( ) [inline], [override]
```

6.30.2 Member Function Documentation

6.30.2.1 on_publication_matched()

```
void DevicePublisher::PubListener::on_publication_matched (
    DataWriter * ,
    const PublicationMatchedStatus & info ) [inline], [override]
```

6.30.3 Member Data Documentation

6.30.3.1 matched_

```
std::atomic_int DevicePublisher::PubListener::matched_
```

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

- [/home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp](#)

6.31 MAX30102::Record Struct Reference

Public Attributes

- uint32_t [red](#) [[STORAGE_SIZE](#)]
- uint32_t [IR](#) [[STORAGE_SIZE](#)]
- uint8_t [head](#)
- uint8_t [tail](#)

6.31.1 Member Data Documentation

6.31.1.1 head

```
uint8_t MAX30102::Record::head
```

6.31.1.2 IR

```
uint32_t MAX30102::Record::IR [STORAGE\_SIZE]
```

6.31.1.3 red

```
uint32_t MAX30102::Record::red [STORAGE\_SIZE]
```

6.31.1.4 tail

```
uint8_t MAX30102::Record::tail
```

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

- [/home/sitcomlab/Projects/VigiSense/src/MAX30102.h](#)

6.32 CircularDelay< type, size >::reverse_iterator Class Reference

```
#include <CircularDelay.hpp>
```

Public Types

- typedef [reverse_iterator](#) [self_type](#)
- typedef std::bidirectional_iterator_tag [iterator_category](#)
- typedef int [difference_type](#)

Public Member Functions

- [reverse_iterator](#) (const [CircularDelay](#)< type, size >::reverse_iterator &it)
- [self_type](#) operator++ ()
- [self_type](#) operator++ (int)
- [self_type](#) operator-- ()
- [self_type](#) operator-- (int)
- type & operator* ()
- type * operator-> ()
- type & operator[] (int index)
- bool operator== (const [self_type](#) &rhs)
- bool operator!= (const [self_type](#) &rhs)

Private Member Functions

- [reverse_iterator](#) (type *data, type *ptr)

Private Attributes

- type * [data_](#) = nullptr
- type * [ptr_](#) = nullptr

Friends

- class [CircularDelay](#)

6.32.1 Member Typedef Documentation

6.32.1.1 difference_type

```
template<typename type , size_t size>
typedef int CircularDelay< type, size >::reverse_iterator::difference_type
```

6.32.1.2 iterator_category

```
template<typename type , size_t size>
typedef std::bidirectional_iterator_tag CircularDelay< type, size >::reverse_iterator::iterator↵
_category
```

6.32.1.3 self_type

```
template<typename type , size_t size>
typedef reverse_iterator CircularDelay< type, size >::reverse_iterator::self_type
```

6.32.2 Constructor & Destructor Documentation

6.32.2.1 reverse_iterator() [1/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::reverse_iterator::reverse_iterator (
    const CircularDelay< type, size >::reverse_iterator & it ) [inline]
```

6.32.2.2 reverse_iterator() [2/2]

```
template<typename type , size_t size>
CircularDelay< type, size >::reverse_iterator::reverse_iterator (
    type * data,
    type * ptr ) [inline], [private]
```

6.32.3 Member Function Documentation

6.32.3.1 operator!=(())

```
template<typename type , size_t size>
bool CircularDelay< type, size >::reverse_iterator::operator!= (
    const self_type & rhs ) [inline]
```

6.32.3.2 operator*()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::reverse_iterator::operator* ( ) [inline]
```

6.32.3.3 operator++() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator++ ( ) [inline]
```

6.32.3.4 operator++() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator++ (
    int ) [inline]
```

6.32.3.5 operator--() [1/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator-- ( ) [inline]
```

6.32.3.6 operator--() [2/2]

```
template<typename type , size_t size>
self_type CircularDelay< type, size >::reverse_iterator::operator-- (
    int ) [inline]
```

6.32.3.7 operator->()

```
template<typename type , size_t size>
type * CircularDelay< type, size >::reverse_iterator::operator-> ( ) [inline]
```

6.32.3.8 operator==()

```
template<typename type , size_t size>
bool CircularDelay< type, size >::reverse_iterator::operator== (
    const self_type & rhs ) [inline]
```

6.32.3.9 operator[]()

```
template<typename type , size_t size>
type & CircularDelay< type, size >::reverse_iterator::operator[] (
    int index ) [inline]
```

6.32.4 Friends And Related Symbol Documentation

6.32.4.1 CircularDelay

```
template<typename type , size_t size>
friend class CircularDelay [friend]
```

6.32.5 Member Data Documentation

6.32.5.1 data_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::reverse_iterator::data_ = nullptr [private]
```

6.32.5.2 ptr_

```
template<typename type , size_t size>
type* CircularDelay< type, size >::reverse_iterator::ptr_ = nullptr [private]
```

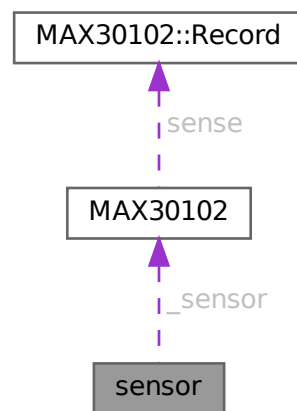
The documentation for this class was generated from the following file:

- [/home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp](#)

6.33 sensor Class Reference

```
#include <Sensor.h>
```

Collaboration diagram for sensor:



Public Member Functions

- [sensor](#) ([MAX30102](#) *[sensor](#))
- [~sensor](#) ()
- void [begin](#) ()
- void [stop](#) ()
- float [getLatestTemperatureF](#) ()
- void [HRcalc](#) ()
- void [stopHRcalc](#) ()
- int [getSpO2](#) ()
- int [getHR](#) ()

Protected Member Functions

- void [loopThread](#) ()
- void [runHRCalculationLoop](#) ()
- void [updateTemperature](#) ()
- void [resetCalculations](#) ()
- int32_t [Derivative](#) (int32_t data)
- int32_t [getPeakThreshold](#) ()
- bool [peakDetect](#) (int32_t data)

Protected Attributes

- [MAX30102](#) * [_sensor](#)
- bool [running](#) = false
- bool [runningHR](#) = false
- int32_t [bpmBuffer](#) [[BPM_BUFFER_SIZE](#)]
- int [nextBPMBufferIndex](#) = 0
- int32_t [spo2Buffer](#) [[SPO2_BUFFER_SIZE](#)]
- int [nextSPO2BufferIndex](#) = 0
- std::chrono::time_point< std::chrono::system_clock > [timeLastLoopRan](#)
- std::chrono::time_point< std::chrono::system_clock > [timeLastIRHeartBeat](#)
- int32_t [irLastValue](#)
- int [latestIRBPM](#)
- int [averageIRBPM](#)
- std::chrono::time_point< std::chrono::system_clock > [timeLastRedHeartBeat](#)
- uint64_t [redLastValue](#)
- int [latestRedBPM](#)
- float [latestTemperature](#) = -999
- int32_t [localMaximalIR](#)
- int32_t [localMinimalIR](#)
- int32_t [localMaximaRed](#)
- int32_t [localMinimaRed](#)
- int32_t [pastMaximasIR](#) [[PAST_PEAKS_SIZE](#)]
- int32_t [pastMinimasIR](#) [[PAST_PEAKS_SIZE](#)]
- int32_t [pastMaximasRed](#) [[PAST_PEAKS_SIZE](#)]
- int32_t [pastMinimasRed](#) [[PAST_PEAKS_SIZE](#)]
- int [R](#)
- int [latestSpO2](#)

Static Protected Attributes

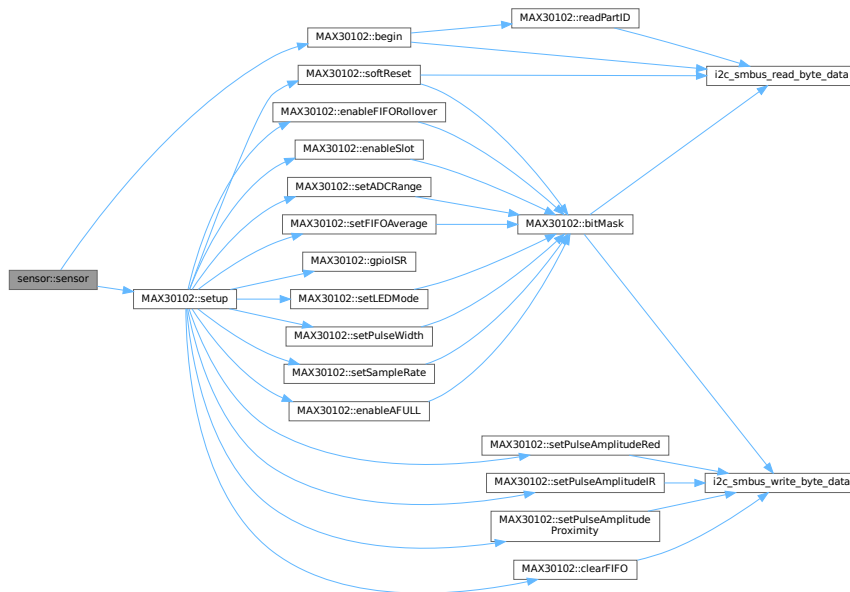
- const static int [BPM_BUFFER_SIZE](#) = 4
- const static int [SPO2_BUFFER_SIZE](#) = 4
- static const int8_t [PAST_PEAKS_SIZE](#) = 2

6.33.1 Constructor & Destructor Documentation

6.33.1.1 sensor()

```
sensor::sensor (
    MAX30102 * s )
```

Constructor to initialize the [MAX30102](#) sensor with the default I2C address and start communication Could also change the class name to "MAX30102Sensor" OR have "MAX30102Sensor" inherit from "sensor". Here is the call graph for this function:



6.33.1.2 ~sensor()

```
sensor::~sensor ( )
```

Here is the call graph for this function:



6.33.2 Member Function Documentation

6.33.2.1 begin()

```
void sensor::begin ( )
```

6.33.2.2 Derivative()

```
int32_t sensor::Derivative (
    int32_t data ) [protected]
```

6.33.2.3 getHR()

```
int sensor::getHR ( )
```

Returns the latest calculated IR heart rate. (unchecked!) Returns the average measured heart rate. This method ignores heart rate values greater than 150 or lower than 45.

6.33.2.4 getLatestTemperatureF()

```
float sensor::getLatestTemperatureF ( )
```

Returns the latest calculated Red heart rate. (unchecked!)

6.33.2.5 getPeakThreshold()

```
int32_t sensor::getPeakThreshold ( ) [protected]
```

Detects peaks in heart data. Returns true when input data is a peak.

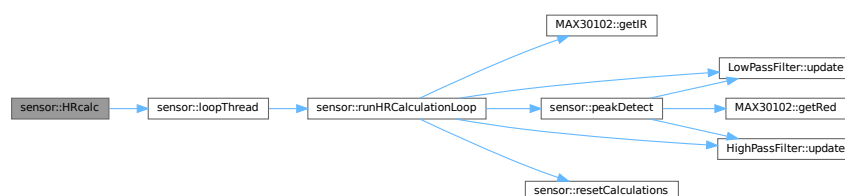
6.33.2.6 getSpO2()

```
int sensor::getSpO2 ( )
```

6.33.2.7 HRcalc()

```
void sensor::HRcalc ( )
```

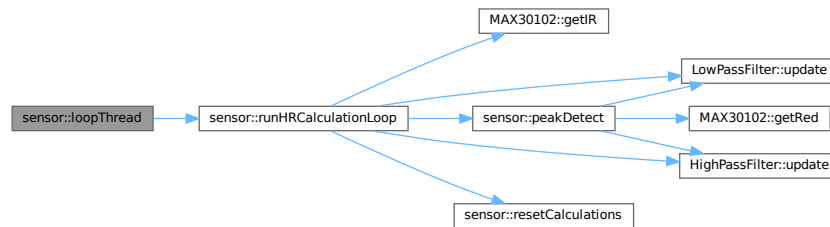
Here is the call graph for this function:



6.33.2.8 loopThread()

```
void sensor::loopThread ( ) [protected]
```

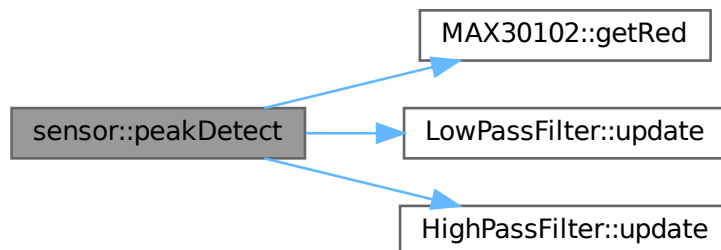
Here is the call graph for this function:



6.33.2.9 peakDetect()

```
bool sensor::peakDetect (
    int32_t data ) [protected]
```

Here is the call graph for this function:



6.33.2.10 resetCalculations()

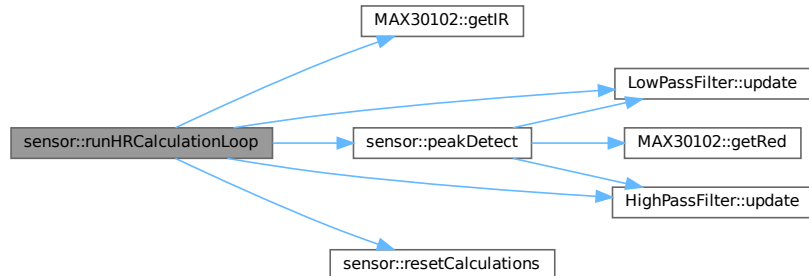
```
void sensor::resetCalculations ( ) [protected]
```

Clears all calculations.

6.33.2.11 runHRCalculationLoop()

```
void sensor::runHRCalculationLoop ( ) [protected]
```

Here is the call graph for this function:



6.33.2.12 stop()

```
void sensor::stop ( )
```

6.33.2.13 stopHRcalc()

```
void sensor::stopHRcalc ( )
```

Stops the calculation loop. You may no longer get heart rate data after calling this function. Here is the call graph for this function:



6.33.2.14 updateTemperature()

```
void sensor::updateTemperature ( ) [protected]
```

Updates the temperature variable. Here is the call graph for this function:



6.33.3 Member Data Documentation

6.33.3.1 `_sensor`

`MAX30102* sensor::_sensor` [protected]

6.33.3.2 `averageIRBPM`

`int sensor::averageIRBPM` [protected]

6.33.3.3 `BPM_BUFFER_SIZE`

`const static int sensor::BPM_BUFFER_SIZE = 4` [static], [protected]

6.33.3.4 `bpmBuffer`

`int32_t sensor::bpmBuffer[BPM_BUFFER_SIZE]` [protected]

6.33.3.5 `irLastValue`

`int32_t sensor::irLastValue` [protected]

6.33.3.6 `latestIRBPM`

`int sensor::latestIRBPM` [protected]

6.33.3.7 `latestRedBPM`

`int sensor::latestRedBPM` [protected]

6.33.3.8 `latestSpO2`

`int sensor::latestSpO2` [protected]

6.33.3.9 `latestTemperature`

`float sensor::latestTemperature = -999` [protected]

6.33.3.10 `localMaximalR`

`int32_t sensor::localMaximalR` [protected]

6.33.3.11 localMaximaRed

```
int32_t sensor::localMaximaRed [protected]
```

6.33.3.12 localMinimaIR

```
int32_t sensor::localMinimaIR [protected]
```

6.33.3.13 localMinimaRed

```
int32_t sensor::localMinimaRed [protected]
```

6.33.3.14 nextBPMBufferIndex

```
int sensor::nextBPMBufferIndex = 0 [protected]
```

6.33.3.15 nextSPO2BufferIndex

```
int sensor::nextSPO2BufferIndex = 0 [protected]
```

6.33.3.16 PAST_PEAKE_SIZE

```
const int8_t sensor::PAST_PEAKE_SIZE = 2 [static], [protected]
```

6.33.3.17 pastMaximasIR

```
int32_t sensor::pastMaximasIR[PAST_PEAKE_SIZE] [protected]
```

6.33.3.18 pastMaximasRed

```
int32_t sensor::pastMaximasRed[PAST_PEAKE_SIZE] [protected]
```

6.33.3.19 pastMinimasIR

```
int32_t sensor::pastMinimasIR[PAST_PEAKE_SIZE] [protected]
```

6.33.3.20 pastMinimasRed

```
int32_t sensor::pastMinimasRed[PAST_PEAKE_SIZE] [protected]
```

6.33.3.21 R

```
int sensor::R [protected]
```

6.33.3.22 redLastValue

```
uint64_t sensor::redLastValue [protected]
```

6.33.3.23 running

```
bool sensor::running = false [protected]
```

6.33.3.24 runningHR

```
bool sensor::runningHR = false [protected]
```

6.33.3.25 SPO2_BUFFER_SIZE

```
const static int sensor::SPO2_BUFFER_SIZE = 4 [static], [protected]
```

6.33.3.26 spo2Buffer

```
int32_t sensor::spo2Buffer[SPO2_BUFFER_SIZE] [protected]
```

6.33.3.27 timeLastIRHeartBeat

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastIRHeartBeat [protected]
```

6.33.3.28 timeLastLoopRan

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastLoopRan [protected]
```

6.33.3.29 timeLastRedHeartBeat

```
std::chrono::time_point<std::chrono::system_clock> sensor::timeLastRedHeartBeat [protected]
```

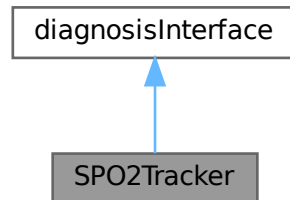
The documentation for this class was generated from the following files:

- /home/sitcomlab/Projects/VigiSense/src/[Sensor.h](#)
- /home/sitcomlab/Projects/VigiSense/src/[Sensor.cpp](#)

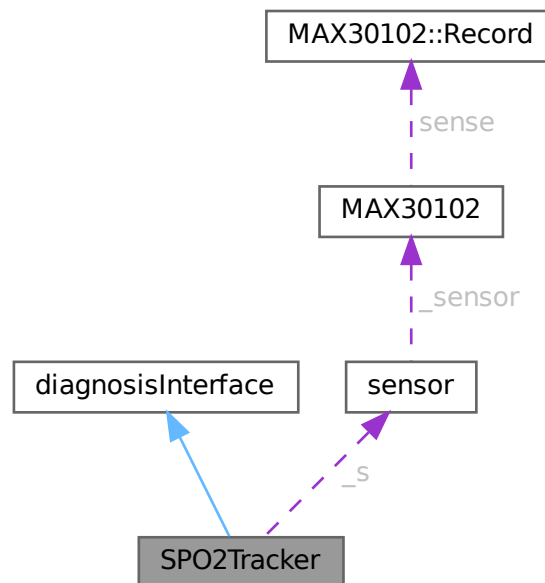
6.34 SPO2Tracker Class Reference

```
#include <SPO2Tracker.h>
```

Inheritance diagram for SPO2Tracker:



Collaboration diagram for SPO2Tracker:



Public Member Functions

- [SPO2Tracker](#) ([sensor](#) *s)
- [~SPO2Tracker](#) ()
- void [start](#) ()
- void [stop](#) ()
- void [ping](#) ()
- int [getVal](#) ()
- void [tracker](#) ()

Protected Member Functions

- void `pingThread` ()

Protected Attributes

- `sensor` * `_s`
- bool `threadRunning` = false
- std::vector< `symptomRange` > `symptomRanges`

Protected Attributes inherited from `diagnosisInterface`

- std::vector< `symptomRange` > `symptomRanges`

Additional Inherited Members**Static Public Member Functions inherited from `diagnosisInterface`**

- static std::string `determineSymptom` (std::vector< `symptomRange` > `symptomRanges`, int val)

6.34.1 Constructor & Destructor Documentation**6.34.1.1 SPO2Tracker()**

```
SPO2Tracker::SPO2Tracker (
    sensor * s )
```

6.34.1.2 ~SPO2Tracker()

```
SPO2Tracker::~~SPO2Tracker ( )
```

Here is the call graph for this function:



6.34.2 Member Function Documentation

6.34.2.1 getVal()

```
int SPO2Tracker::getVal ( ) [virtual]
```

Implements [diagnosisInterface](#).

Here is the call graph for this function:



6.34.2.2 ping()

```
void SPO2Tracker::ping ( ) [virtual]
```

Implements [diagnosisInterface](#).

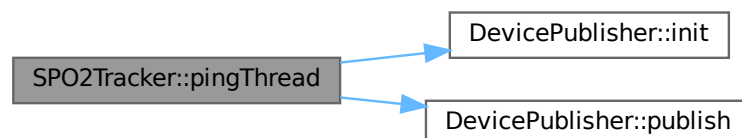
Here is the call graph for this function:



6.34.2.3 pingThread()

```
void SPO2Tracker::pingThread ( ) [protected]
```

Here is the call graph for this function:

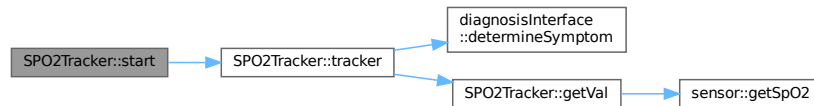


6.34.2.4 start()

```
void SPO2Tracker::start ( ) [virtual]
```

Implements [diagnosisInterface](#).

Here is the call graph for this function:



6.34.2.5 stop()

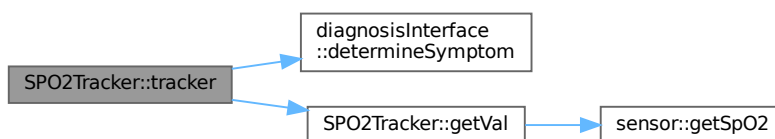
```
void SPO2Tracker::stop ( ) [virtual]
```

Implements [diagnosisInterface](#).

6.34.2.6 tracker()

```
void SPO2Tracker::tracker ( )
```

Here is the call graph for this function:



6.34.3 Member Data Documentation

6.34.3.1 _s

```
sensor* SPO2Tracker::_s [protected]
```

6.34.3.2 symptomRanges

```
std::vector<symptomRange> SPO2Tracker::symptomRanges [protected]
```

Initial value:

```
{  
    {0,88,"Critically Low Oxygen concentration"},  
    {88,92,"Concerningly Low Oxygen Concentration"},  
    {92,100,"Healthy Oxygen Concentration"}}}
```

6.34.3.3 threadRunning

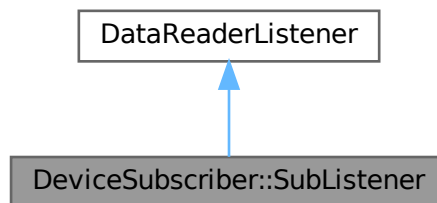
```
bool SPO2Tracker::threadRunning = false [protected]
```

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

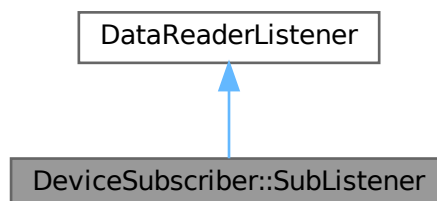
- [/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h](#)
- [/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp](#)

6.35 DeviceSubscriber::SubListener Class Reference

Inheritance diagram for DeviceSubscriber::SubListener:



Collaboration diagram for DeviceSubscriber::SubListener:



Public Member Functions

- [SubListener](#) ()
- [~SubListener](#) () override
- void [on_subscription_matched](#) (DataReader *, const SubscriptionMatchedStatus &info) override
- void [on_data_available](#) (DataReader *reader) override

6.35.1 Constructor & Destructor Documentation**6.35.1.1 SubListener()**

```
DeviceSubscriber::SubListener::SubListener ( ) [inline]
```

6.35.1.2 ~SubListener()

```
DeviceSubscriber::SubListener::~~SubListener ( ) [inline], [override]
```

6.35.2 Member Function Documentation**6.35.2.1 on_data_available()**

```
void DeviceSubscriber::SubListener::on_data_available (
    DataReader * reader ) [inline], [override]
```

6.35.2.2 on_subscription_matched()

```
void DeviceSubscriber::SubListener::on_subscription_matched (
    DataReader * ,
    const SubscriptionMatchedStatus & info ) [inline], [override]
```

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

- [/home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp](#)

6.36 symptomRange Struct Reference

```
#include <DiagnosisInterface.h>
```

Public Attributes

- float [min](#)
- float [max](#)
- std::string [symptom](#)

6.36.1 Member Data Documentation

6.36.1.1 max

`float symptomRange::max`

6.36.1.2 min

`float symptomRange::min`

6.36.1.3 symptom

`std::string symptomRange::symptom`

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

- </home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h>

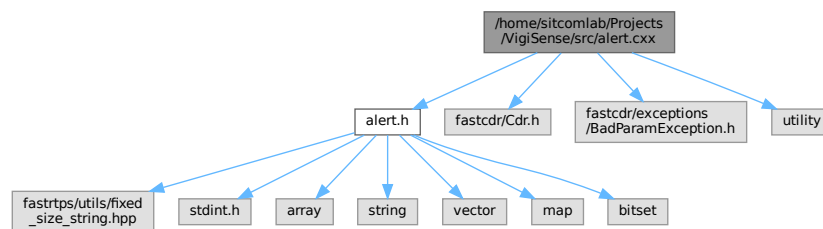
Chapter 7

File Documentation

7.1 /home/sitcomlab/Projects/VigiSense/src/alert.cxx File Reference

```
#include "alert.h"  
#include <fastcdr/Cdr.h>  
#include <fastcdr/exceptions/BadParamException.h>  
#include <utility>
```

Include dependency graph for alert.cxx:

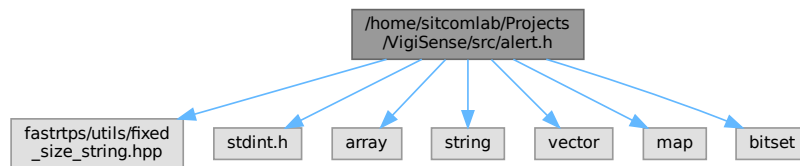


7.2 /home/sitcomlab/Projects/VigiSense/src/alert.h File Reference

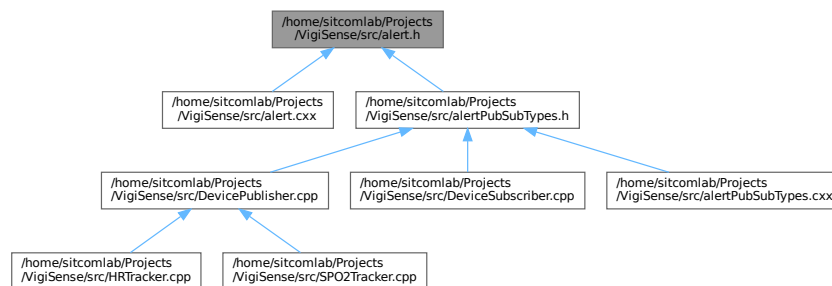
```
#include <fastrtps/utils/fixed_size_string.hpp>  
#include <stdint.h>  
#include <array>  
#include <string>  
#include <vector>  
#include <map>
```

```
#include <bitset>
```

Include dependency graph for alert.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [alert](#)

This class represents the structure alert defined by the user in the IDL file.

Namespaces

- namespace [eprosima](#)
- namespace [eprosima::fastcdr](#)

Macros

- `#define` [eProsima_user_DllExport](#)
- `#define` [alert_DllAPI](#)

7.2.1 Detailed Description

This header file contains the declaration of the described types in the IDL file.

This file was generated by the tool gen.

7.2.2 Macro Definition Documentation

7.2.2.1 alert_DllAPI

```
#define alert_DllAPI
```

7.2.2.2 eProxima_user_DllExport

```
#define eProxima_user_DllExport
```

7.3 alert.h

[Go to the documentation of this file.](#)

```
00001 // Copyright 2016 Proyectos y Sistemas de Mantenimiento SL (eProxima).
00002 //
00003 // Licensed under the Apache License, Version 2.0 (the "License");
00004 // you may not use this file except in compliance with the License.
00005 // You may obtain a copy of the License at
00006 //
00007 //     http://www.apache.org/licenses/LICENSE-2.0
00008 //
00009 // Unless required by applicable law or agreed to in writing, software
00010 // distributed under the License is distributed on an "AS IS" BASIS,
00011 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
00012 // See the license for the specific language governing permissions and
00013 // limitations under the License.
00014
00022 #ifndef _FAST_DDS_GENERATED_ALERT_H_
00023 #define _FAST_DDS_GENERATED_ALERT_H_
00024
00025
00026 #include <fastrtps/utils/fixed_size_string.hpp>
00027
00028 #include <stdint.h>
00029 #include <array>
00030 #include <string>
00031 #include <vector>
00032 #include <map>
00033 #include <bitset>
00034
00035 #if defined(_WIN32)
00036 #if defined(EPROSIMA_USER_DLL_EXPORT)
00037 #define eProxima_user_DllExport __declspec( dllexport )
00038 #else
00039 #define eProxima_user_DllExport
00040 #endif // EPROSIMA_USER_DLL_EXPORT
00041 #else
00042 #define eProxima_user_DllExport
00043 #endif // _WIN32
00044
00045 #if defined(_WIN32)
00046 #if defined(EPROSIMA_USER_DLL_EXPORT)
00047 #if defined(alert_SOURCE)
00048 #define alert_DllAPI __declspec( dllexport )
00049 #else
00050 #define alert_DllAPI __declspec( dllimport )
00051 #endif // alert_SOURCE
00052 #else
00053 #define alert_DllAPI
00054 #endif // EPROSIMA_USER_DLL_EXPORT
00055 #else
00056 #define alert_DllAPI
00057 #endif // _WIN32
00058
00059 namespace eproxima {
00060 namespace fastcdr {
00061 class Cdr;
00062 } // namespace fastcdr
00063 } // namespace eproxima
00064
00065
00070 class alert
00071 {
```

```

00072 public:
00073
00077     eProsima_user_DllExport alert();
00078
00082     eProsima_user_DllExport ~alert();
00083
00088     eProsima_user_DllExport alert(
00089         const alert& x);
00090
00095     eProsima_user_DllExport alert(
00096         alert&& x) noexcept;
00097
00102     eProsima_user_DllExport alert& operator =(
00103         const alert& x);
00104
00109     eProsima_user_DllExport alert& operator =(
00110         alert&& x) noexcept;
00111
00116     eProsima_user_DllExport bool operator ==(
00117         const alert& x) const;
00118
00123     eProsima_user_DllExport bool operator !=(
00124         const alert& x) const;
00125
00130     eProsima_user_DllExport void index(
00131         uint32_t _index);
00132
00137     eProsima_user_DllExport uint32_t index() const;
00138
00143     eProsima_user_DllExport uint32_t& index();
00144
00149     eProsima_user_DllExport void message(
00150         const std::string& _message);
00151
00156     eProsima_user_DllExport void message(
00157         std::string&& _message);
00158
00163     eProsima_user_DllExport const std::string& message() const;
00164
00169     eProsima_user_DllExport std::string& message();
00170
00177     eProsima_user_DllExport static size_t getMaxCdrSerializedSize(
00178         size_t current_alignment = 0);
00179
00186     eProsima_user_DllExport static size_t getCdrSerializedSize(
00187         const alert& data,
00188         size_t current_alignment = 0);
00189
00190
00195     eProsima_user_DllExport void serialize(
00196         eprosima::fastcdr::Cdr& cdr) const;
00197
00202     eProsima_user_DllExport void deserialize(
00203         eprosima::fastcdr::Cdr& cdr);
00204
00205
00206
00213     eProsima_user_DllExport static size_t getKeyMaxCdrSerializedSize(
00214         size_t current_alignment = 0);
00215
00219     eProsima_user_DllExport static bool isKeyDefined();
00220
00225     eProsima_user_DllExport void serializeKey(
00226         eprosima::fastcdr::Cdr& cdr) const;
00227
00228 private:
00229
00230     uint32_t m_index;
00231     std::string m_message;
00232 };
00233
00234 #endif // _FAST_DDS_GENERATED_ALERT_H_

```

7.4 /home/sitcomlab/Projects/VigiSense/src/alert.idl File Reference

Classes

- struct [alert](#)

This class represents the structure alert defined by the user in the IDL file.

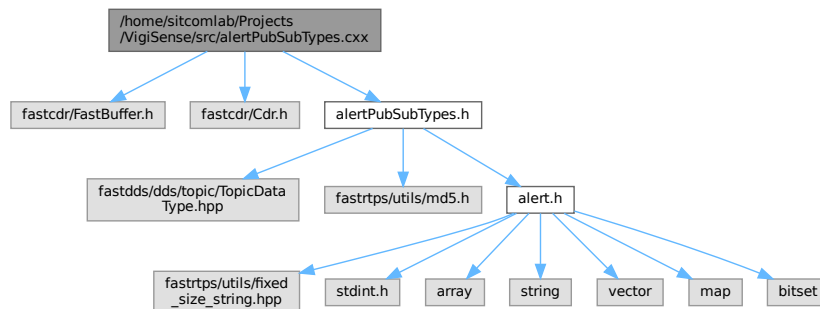
7.5 alert.idl

[Go to the documentation of this file.](#)

```
00001 struct alert
00002 {
00003     unsigned long index;
00004     string message;
00005 };
```

7.6 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx File Reference

```
#include <fastcdr/FastBuffer.h>
#include <fastcdr/Cdr.h>
#include "alertPubSubTypes.h"
Include dependency graph for alertPubSubTypes.cxx:
```



Typedefs

- using `SerializedPayload_t` = `eprosima::fastrtps::rtps::SerializedPayload_t`
- using `InstanceHandle_t` = `eprosima::fastrtps::rtps::InstanceHandle_t`

7.6.1 Typedef Documentation

7.6.1.1 InstanceHandle_t

```
using InstanceHandle_t = eprosima::fastrtps::rtps::InstanceHandle_t
```

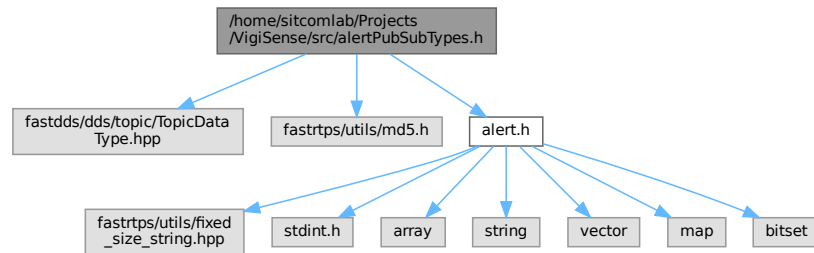
7.6.1.2 SerializedPayload_t

```
using SerializedPayload_t = eprosima::fastrtps::rtps::SerializedPayload_t
```

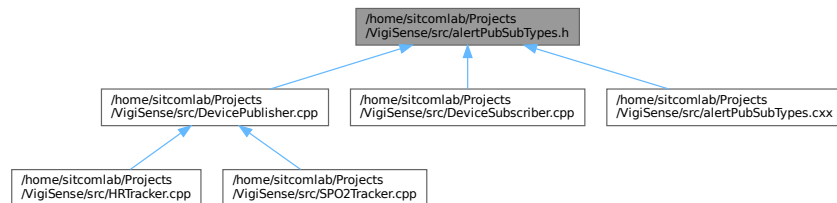
7.7 /home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h File Reference

```
#include <fastdds/dds/topic/TopicDataType.hpp>
#include <fastrtps/utils/md5.h>
#include "alert.h"
```

Include dependency graph for alertPubSubTypes.h:



This graph shows which files directly or indirectly include this file:



Classes

- class `alertPubSubType`

This class represents the `TopicDataType` of the type `alert` defined by the user in the IDL file.

7.7.1 Detailed Description

This header file contains the declaration of the serialization functions.

This file was generated by the tool `fastcdrngen`.

7.8 alertPubSubTypes.h

[Go to the documentation of this file.](#)

```

00001 // Copyright 2016 Proyectos y Sistemas de Mantenimiento SL (eProsima).
00002 //
00003 // Licensed under the Apache License, Version 2.0 (the "License");
00004 // you may not use this file except in compliance with the License.
00005 // You may obtain a copy of the License at
00006 //
00007 //     http://www.apache.org/licenses/LICENSE-2.0
00008 //
00009 // Unless required by applicable law or agreed to in writing, software
00010 // distributed under the License is distributed on an "AS IS" BASIS,
00011 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
00012 // See the License for the specific language governing permissions and
00013 // limitations under the License.
00014
00023 #ifndef _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_
00024 #define _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_
00025
00026 #include <fastdds/dds/topic/TopicDataType.hpp>
00027 #include <fastrtps/utils/md5.h>
00028
00029 #include "alert.h"
00030
00031 #if !defined(GEN_API_VER) || (GEN_API_VER != 1)
00032 #error \
00033     Generated alert is not compatible with current installed Fast DDS. Please, regenerate it with
00034     fastddsgen.
00035 #endif // GEN_API_VER
00036
00040 class alertPubSubType : public eprosima::fastdds::dds::TopicDataType
00041 {
00042 public:
00043
00044     typedef alert type;
00045
00046     eProsima_user_DllExport alertPubSubType();
00047
00048     eProsima_user_DllExport virtual ~alertPubSubType() override;
00049
00050     eProsima_user_DllExport virtual bool serialize(
00051         void* data,
00052         eprosima::fastrtps::rtps::SerializedPayload_t* payload) override;
00053
00054     eProsima_user_DllExport virtual bool deserialize(
00055         eprosima::fastrtps::rtps::SerializedPayload_t* payload,
00056         void* data) override;
00057
00058     eProsima_user_DllExport virtual std::function<uint32_t()> getSerializedSizeProvider(
00059         void* data) override;
00060
00061     eProsima_user_DllExport virtual bool getKey(
00062         void* data,
00063         eprosima::fastrtps::rtps::InstanceHandle_t* ihandle,
00064         bool force_md5 = false) override;
00065
00066     eProsima_user_DllExport virtual void* createData() override;
00067
00068     eProsima_user_DllExport virtual void deleteData(
00069         void* data) override;
00070
00071 #ifdef TOPIC_DATA_TYPE_API_HAS_IS_BOUNDED
00072     eProsima_user_DllExport inline bool is_bounded() const override
00073     {
00074         return false;
00075     }
00076 #endif // TOPIC_DATA_TYPE_API_HAS_IS_BOUNDED
00077
00078 #ifdef TOPIC_DATA_TYPE_API_HAS_IS_PLAIN
00079     eProsima_user_DllExport inline bool is_plain() const override
00080     {
00081         return false;
00082     }
00083 #endif // TOPIC_DATA_TYPE_API_HAS_IS_PLAIN
00084
00085 #ifdef TOPIC_DATA_TYPE_API_HAS_IS_PLAIN
00086 #endif
00087 #ifdef TOPIC_DATA_TYPE_API_HAS_CONSTRUCT_SAMPLE
00088     eProsima_user_DllExport inline bool construct_sample(
00089         void* memory) const override
00090     {
00091         (void)memory;
00092         return false;
00093     }
00094 #endif

```

```

00094
00095 #endif // TOPIC_DATA_TYPE_API_HAS_CONSTRUCT_SAMPLE
00096
00097     MD5 m_md5;
00098     unsigned char* m_keyBuffer;
00099 };
00100
00101 #endif // _FAST_DDS_GENERATED_ALERT_PUBSUBTYPES_H_

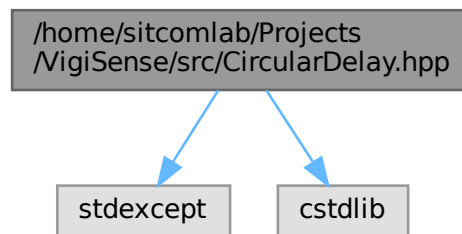
```

7.9 /home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp File Reference

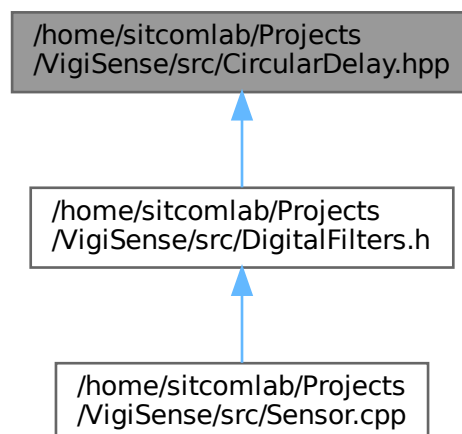
```
#include <stdexcept>
```

```
#include <cstdlib>
```

Include dependency graph for CircularDelay.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class `CircularDelay< type, size >`
A class that functions as a sample buffer.
- class `CircularDelay< type, size >::iterator`
- class `CircularDelay< type, size >::const_iterator`
- class `CircularDelay< type, size >::reverse_iterator`
- class `CircularDelay< type, size >::const_reverse_iterator`

7.10 CircularDelay.hpp

[Go to the documentation of this file.](#)

```

00001
00020 #include <stdexcept>
00021 #include <cstdlib>
00030 template<typename type, size_t size>
00031 class CircularDelay{
00032 public:
00033     CircularDelay();
00034     class iterator{
00035     public:
00036         typedef iterator self_type;
00037         typedef std::bidirectional_iterator_tag iterator_category;
00038         typedef int difference_type;
00039         iterator(const CircularDelay<type, size>::iterator& it):data_(it.data_), ptr_(it.ptr_){}
00040         self_type operator++() {
00041             if(++ptr_ == data_ + size + 1){
00042                 ptr_ = data_;
00043             }
00044             return *this;
00045         }
00046         self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
00047         self_type operator--() {
00048             if(ptr_ == data_){
00049                 ptr_ = data_ + size;
00050             }
00051             else{
00052                 --ptr_;
00053             }
00054             return *this;
00055         }
00056         self_type operator--(int) {self_type ret = *this; --*this; return ret;}
00057         type& operator*() { return *ptr_; }
00058         type* operator->() { return ptr_; }
00059         type& operator[](unsigned int index){
00060             // Note that ptr_ - data_ is converting ptr_ to index value;
00061             index += ptr_ - data_;
00062             if(index >= size + 1){
00063                 index -= size + 1;
00064             }
00065             return data_[index];
00066         }
00067         bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
00068         bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
00069     private:
00070         iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
00071         friend class CircularDelay;
00072         type* data_ = nullptr;
00073         type* ptr_ = nullptr;
00074     };
00075     class const_iterator{
00076     public:
00077         typedef const_iterator self_type;
00078         typedef std::bidirectional_iterator_tag iterator_category;
00079         typedef int difference_type;
00080         const_iterator(const CircularDelay<type, size>::const_iterator& it):data_(it.data_),
ptr_(it.ptr_){}
00081         self_type operator++() {
00082             if(++ptr_ == data_ + size + 1){
00083                 ptr_ = data_;
00084             }
00085             return *this;
00086         }
00087         self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
00088         self_type operator--() {
00089             if(ptr_ == data_){
00090                 ptr_ = data_ + size;

```

```

00091         }
00092         else{
00093             --ptr_;
00094         }
00095         return *this;
00096     }
00097     self_type operator--(int) {self_type ret = *this; --*this; return ret;}
00098     const type& operator*() { return *ptr_; }
00099     const type* operator->() { return ptr_; }
00100     const type& operator[](unsigned int index){
00101         // Note that ptr_ - data_ is converting ptr_ to index value;
00102         index += ptr_ - data_;
00103         if(index >= size + 1){
00104             index -= size + 1;
00105         }
00106         return data_[index];
00107     }
00108     bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
00109     bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
00110 private:
00111     const_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
00112     friend class CircularDelay;
00113     type* data_ = nullptr;
00114     type* ptr_ = nullptr;
00115 };
00116 class reverse_iterator{
00117 public:
00118     typedef reverse_iterator self_type;
00119     typedef std::bidirectional_iterator_tag iterator_category;
00120     typedef int difference_type;
00121     reverse_iterator(const CircularDelay<type, size>::reverse_iterator& it):data_(it.data_),
00122 ptr_(it.ptr_){}
00123     self_type operator++() {
00124         if(ptr_ == data_){
00125             ptr_ = data_ + size;
00126         }
00127         else{
00128             --ptr_;
00129         }
00130         return *this;
00131     }
00132     self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
00133     self_type operator--() {
00134         if(++ptr_ == data_ + size + 1){
00135             ptr_ = data_;
00136         }
00137         return *this;
00138     }
00139     self_type operator--(int) {self_type ret = *this; --*this; return ret;}
00140     type& operator*() { return *ptr_; }
00141     type* operator->() { return ptr_; }
00142     type& operator[](int index){
00143         // Convert ptr_ to index value;
00144         index = ptr_ - data_ - index;
00145         if(index < 0){
00146             index += size + 1;
00147         }
00148         return data_[index];
00149     }
00150     bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
00151     bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
00152 private:
00153     reverse_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
00154     friend class CircularDelay;
00155     type* data_ = nullptr;
00156     type* ptr_ = nullptr;
00157 };
00158 class const_reverse_iterator{
00159 public:
00160     typedef const_reverse_iterator self_type;
00161     typedef std::bidirectional_iterator_tag iterator_category;
00162     typedef int difference_type;
00163     const_reverse_iterator(const CircularDelay<type, size>::const_reverse_iterator&
00164 it):data_(it.data_), ptr_(it.ptr_){}
00165     self_type operator++() {
00166         if(ptr_ == data_){
00167             ptr_ = data_ + size;
00168         }
00169         else{
00170             --ptr_;
00171         }
00172         return *this;
00173     }
00174     self_type operator++(int) {self_type ret = *this; ++*this; return ret;}
00175     self_type operator--() {
00176         if(++ptr_ == data_ + size + 1){
00177             ptr_ = data_;

```



```

00176         }
00177         return *this;
00178     }
00179     self_type operator--(int) {self_type ret = *this; --*this; return ret;}
00180     const type& operator*() { return *ptr_; }
00181     const type* operator->() { return ptr_; }
00182     const type& operator[](int index){
00183         // Convert ptr_ to index value;
00184         index = ptr_ - data_ - index;
00185         if(index < 0){
00186             index += size + 1;
00187         }
00188         return data_[index];
00189     }
00190     bool operator==(const self_type& rhs) { return ptr_ == rhs.ptr_; }
00191     bool operator!=(const self_type& rhs) { return ptr_ != rhs.ptr_; }
00192 private:
00193     const_reverse_iterator(type* data, type* ptr) : data_(data), ptr_(ptr) { }
00194     friend class CircularDelay;
00195     type* data_ = nullptr;
00196     type* ptr_ = nullptr;
00197 };
00198 type push(type input);
00199 type get(size_t delay);
00200 iterator end(){return setIterator;}
00201 iterator begin(){
00202     iterator it(setIterator);
00203     ++it;
00204     return it;
00205 }
00206 reverse_iterator rend(){return reverse_iterator(data, setIterator.ptr_);}
00207 reverse_iterator rbegin(){
00208     reverse_iterator it(data, setIterator.ptr_);
00209     ++it;
00210     return it;
00211 }
00212 private:
00213     type data[size + 1];
00214     iterator setIterator = iterator(data, data);
00215 };
00216
00223 template<typename type, size_t size>
00224 CircularDelay<type, size>::CircularDelay(){
00225     for (size_t i = 0; i < size + 1; ++i){
00226         data[i] = 0;
00227     }
00228 }
00229
00240 template<typename type, size_t size>
00241 type CircularDelay<type, size>::push(type input){
00242     *setIterator = input;
00243     setIterator++;
00244     return input;
00245 }
00246
00258 template<typename type, size_t size>
00259 type CircularDelay<type, size>::get(size_t delay){
00260     if(delay >= size + 1)
00261         throw(std::domain_error("Tried to get a value that is longer ago than the size of a
CircularDelay."));
00262     reverse_iterator itRbegin(rbegin());
00263     return itRbegin[delay];
00264 }

```

7.11 /home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp File Reference

```

#include "alertPubSubTypes.h"
#include <chrono>
#include <thread>
#include <fastdds/dds/domain/DomainParticipant.hpp>
#include <fastdds/dds/domain/DomainParticipantFactory.hpp>
#include <fastdds/dds/publisher/DataWriter.hpp>
#include <fastdds/dds/publisher/DataWriterListener.hpp>
#include <fastdds/dds/publisher/Publisher.hpp>

```



```

00033
00034 class DevicePublisher
00035 {
00036 private:
00037
00038     DomainParticipant* participant_ = nullptr;
00039
00040     Publisher* publisher_ = nullptr;
00041
00042     Topic* topic_ = nullptr;
00043
00044     DataWriter* writer_ = nullptr;
00045
00046     TypeSupport type_;
00047
00048     class PubListener : public DataWriterListener
00049     {
00050     public:
00051
00052         PubListener()
00053             : matched_(0)
00054         {
00055         }
00056
00057         ~PubListener() override
00058         {
00059         }
00060
00061         void on_publication_matched(
00062             DataWriter*,
00063             const PublicationMatchedStatus& info) override
00064         {
00065             if (info.current_count_change == 1)
00066             {
00067                 matched_ = info.total_count;
00068                 std::cout << "Publisher matched." << std::endl;
00069             }
00070             else if (info.current_count_change == -1)
00071             {
00072                 matched_ = info.total_count;
00073                 std::cout << "Publisher unmatched." << std::endl;
00074             }
00075             else
00076             {
00077                 std::cout << info.current_count_change
00078                     << " is not a valid value for PublicationMatchedStatus current count change." <<
00079                 std::endl;
00080             }
00081
00082             std::atomic_int matched_;
00083
00084         } listener_;
00085
00086     public:
00087
00088         DevicePublisher() : type_(new alertPubSubType()) {}
00089
00090         virtual ~DevicePublisher()
00091         {
00092             if (writer_ != nullptr)
00093             {
00094                 publisher_>delete_datawriter(writer_);
00095             }
00096             if (publisher_ != nullptr)
00097             {
00098                 participant_>delete_publisher(publisher_);
00099             }
00100             if (topic_ != nullptr)
00101             {
00102                 participant_>delete_topic(topic_);
00103             }
00104             DomainParticipantFactory::get_instance()->delete_participant(participant_);
00105         }
00106
00107         bool init()
00108         {
00109             DomainParticipantQos participantQos;
00110             participantQos.name("Participant_publisher");
00111             participant_ = DomainParticipantFactory::get_instance()->create_participant(0,
00112                 participantQos);
00113
00114             if (participant_ == nullptr)
00115             {
00116                 return false;
00117             }
00118

```

```

00119         // Register the Type
00120         type_.register_type(participant_);
00121
00122         // Create the publications Topic
00123         // !! Important that this matches with the name of message defined in HelloWorldMsg.idl !!
00124         topic_ = participant_>create_topic("HelloWorldTopic", "alert", TOPIC_QOS_DEFAULT);
00125
00126         if (topic_ == nullptr)
00127         {
00128             return false;
00129         }
00130
00131         // Create the Publisher
00132         publisher_ = participant_>create_publisher(PUBLISHER_QOS_DEFAULT, nullptr);
00133
00134         if (publisher_ == nullptr)
00135         {
00136             return false;
00137         }
00138
00139         // Create the DataWriter
00140         writer_ = publisher_>create_datawriter(topic_, DATAWRITER_QOS_DEFAULT, &listener_);
00141
00142         if (writer_ == nullptr)
00143         {
00144             return false;
00145         }
00146         return true;
00147     }
00148
00150     bool publish(alert& hello)
00151     {
00152         if (listener_.matched_ > 0)
00153         {
00154             writer_>write(&hello);
00155             return true;
00156         }
00157         return false;
00158     }
00159
00160 };

```

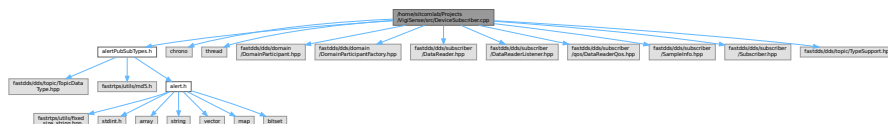
7.13 /home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp File Reference

```

#include "alertPubSubTypes.h"
#include <chrono>
#include <thread>
#include <fastdds/dds/domain/DomainParticipant.hpp>
#include <fastdds/dds/domain/DomainParticipantFactory.hpp>
#include <fastdds/dds/subscriber/DataReader.hpp>
#include <fastdds/dds/subscriber/DataReaderListener.hpp>
#include <fastdds/dds/subscriber/qos/DataReaderQos.hpp>
#include <fastdds/dds/subscriber/SampleInfo.hpp>
#include <fastdds/dds/subscriber/Subscriber.hpp>
#include <fastdds/dds/topic/TypeSupport.hpp>

```

Include dependency graph for DeviceSubscriber.cpp:



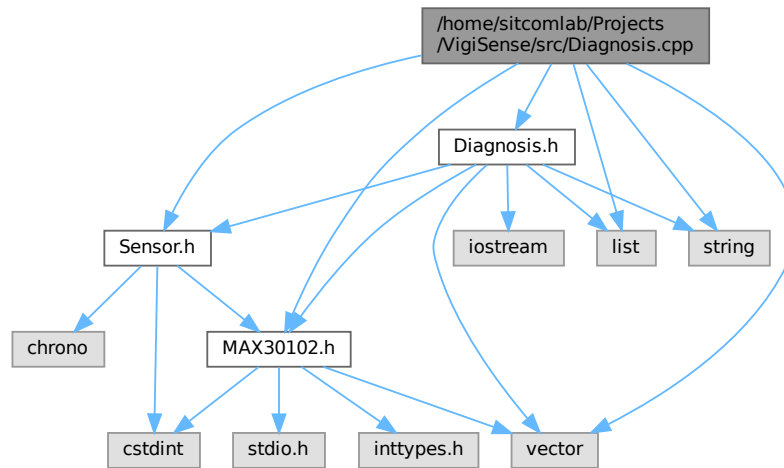
Classes

- class [DeviceSubscriber](#)
- class [DeviceSubscriber::SubListener](#)

7.14 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp File Reference

```
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
#include "Sensor.h"
#include "Diagnosis.h"
```

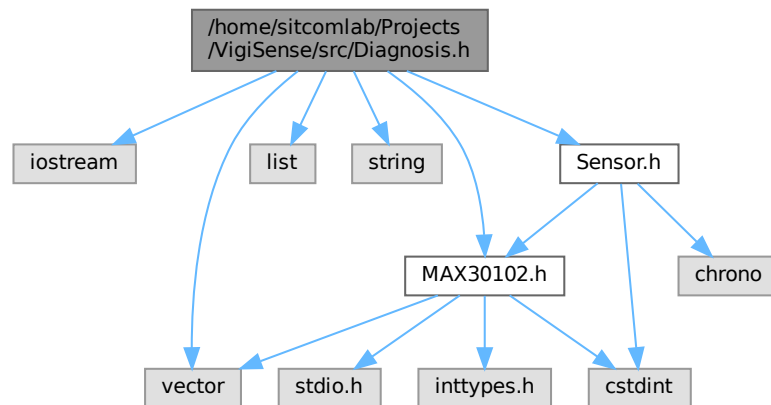
Include dependency graph for Diagnosis.cpp:



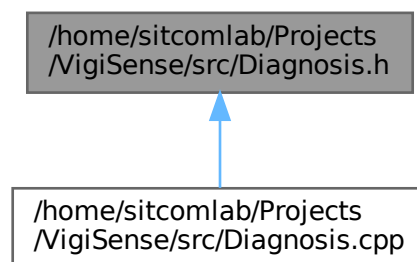
7.15 /home/sitcomlab/Projects/VigiSense/src/Diagnosis.h File Reference

```
#include <iostream>
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"
#include "Sensor.h"
```

Include dependency graph for Diagnosis.h:



This graph shows which files directly or indirectly include this file:



Classes

- class `diagnosis`
- struct `diagnosis::DiagnosisRange`
- struct `diagnosis::DiagnosesTable`

7.16 Diagnosis.h

[Go to the documentation of this file.](#)

```

00001 #include <iostream>
00002 #include <vector>
00003 #include <list>
00004 #include <string>
00005 #include "MAX30102.h"
00006 #include "Sensor.h"

```

```

00007
00008 //This is going to be an abstract class that will be inherited by the diagnosis classes for each
    biosignal
00009
00010 class diagnosis {
00011 public:
00012     // Public member functions and variables
00013
00014     //Struct for the individual diagnosis range
00015     typedef struct DiagnosisRange{
00016         float min;
00017         float max;
00018         std::string diagnosis;
00019     } Diagnosis_Range;
00020
00021     Diagnosis_Range stdDiagnosis; // Creates a standard symptom range for the diagnosis
00022     Diagnosis_Range CustomDiagnosis; //Creates a custom symptom range for the diagnosis
00023
00024     //Struct for the diagnoses table
00025     typedef struct DiagnosesTable {
00026         std::vector<DiagnosisRange> Diagnoses; //Contains all the possible diagnoses for the biosignal
00027     } _Diagnoses;
00028
00029     _Diagnoses stdDiagnoses; //Creates a standard diagnoses table for the biosignal
00030     _Diagnoses CustomDiagnoses; //Creates a custom diagnoses table for the biosignal
00031
00032     //Function to set customized symptom ranges for the biosignal
00033     void SetdiagnosisRanges(float minimum, float maximum, std::string diagnosis); //Sets the symptom
        ranges for the biosignal
00034
00035     //Function to determine the symptom based on the biosignal value
00036     std::string determineDiagnosis() { return ""; }
00037
00038     void critCheck() {} //Checks if the biosignal value crosses the critical value and initiates an
        immediate alert
00039
00040     void findMinMax() {} //Finds the minimum and maximum values in the diagnoses table
00041
00042     float CriticalLow; //Variable to store the critical low value
00043     float CriticalHigh; //Variable to store the critical high value
00044
00045     //function to display the diagnosis
00046     virtual void displayDiagnosis(); //Pure virtual function to display the diagnosis
00047
00048     //function for critical value crossing
00049     virtual void critRangeAlert(); //Pure virtual function for critical value crossing alert
00050 private:
00051     // Private member functions and variables
00052
00053 };

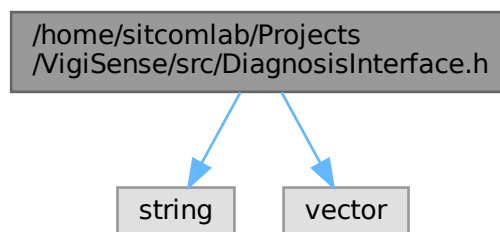
```

7.17 /home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h File Reference

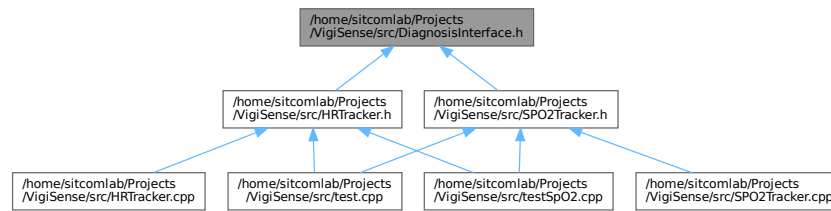
```
#include <string>
```

```
#include <vector>
```

Include dependency graph for DiagnosisInterface.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [symptomRange](#)
- class [diagnosisInterface](#)

7.18 DiagnosisInterface.h

[Go to the documentation of this file.](#)

```

00001 #pragma once
00002 #include <string>
00003 #include <vector>
00004
00005 struct symptomRange{
00006     float min;
00007     float max;
00008     std::string symptom;
00009 };
00010
00011 class diagnosisInterface {
00012     public:
00013         virtual void start() = 0;
00014         virtual void stop() = 0;
00015         virtual void ping() = 0;
00016         virtual int getVal() = 0;
00017
00018         static std::string determineSymptom(std::vector<symptomRange> symptomRanges, int val){
00019             for (int i = 0; i < symptomRanges.size(); ++i){
00020                 if (val>symptomRanges[i].min && val<symptomRanges[i].max){
00021                     return symptomRanges[i].symptom;
00022                 }
00023             }
00024             if (val < symptomRanges[0].min) {
00025                 return "critLow";
00026             } else if (val > symptomRanges[symptomRanges.size() -1].max)
00027             {
00028                 return "critHigh";
00029             } else {
00030                 return "Out of range";
00031             }
00032         };
00033
00034     protected:
00035         std::vector<symptomRange> symptomRanges;
00036 };
00037 
```

7.19 /home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h File Reference

```

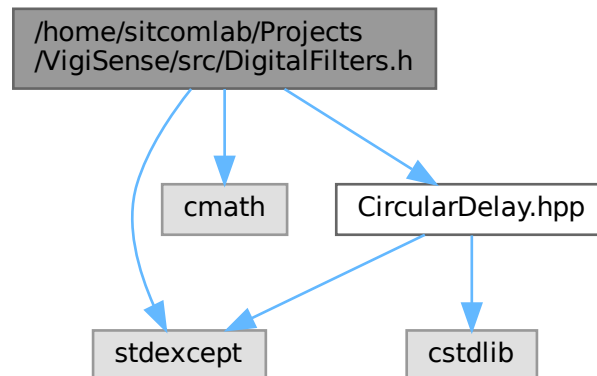
#include <stdexcept>
#include <cmath>

```

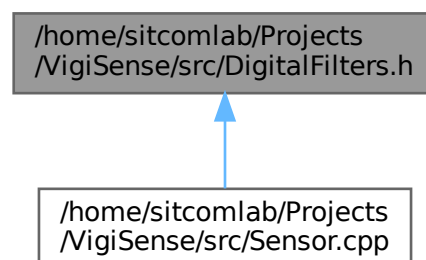


```
#include "CircularDelay.hpp"
```

Include dependency graph for DigitalFilters.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [DigitalFilter< Type >](#)
Abstract base class for digital moving filters.
- class [Differentiator< T >](#)
Class for differentiator.
- class [LowPassFilter](#)
Class for a low pass filter.
- class [LowPassFilter2](#)
Class for a 2nd order low pass filter.
- class [HighPassFilter](#)
Class for high pass filter using bilinear transform.
- class [HighPassFilter3](#)

Class for third order high pass filter. This is designed using the bilinear transform.

- class [LowPassFilter3](#)

Class for third order high pass filter. This is designed using the bilinear transform.

- class [LowPassFilter3MatchedZ](#)

Class for third order high pass filter. This is designed using the matched Z transform.

- class [LowPassFilter3DiffApprox](#)

*Class for third order high pass filter. This is designed using the approximated differtial approuch where $s=(Z-1)/(Z*T)$.*

- class [MovingAvarageFilter< size >](#)

Namespaces

- namespace [tps](#)

Functions

- `template<typename T >`
`constexpr T squareOf (T input)`
- `template<typename T >`
`constexpr T tps::pow (T input, unsigned int power)`
- `template<typename T >`
`constexpr T calcC_Cr (T k, T m)`

7.19.1 Function Documentation

7.19.1.1 [calcC_Cr\(\)](#)

```
template<typename T >
constexpr T calcC_Cr (
    T k,
    T m ) [constexpr]
```

7.19.1.2 [squareOf\(\)](#)

```
template<typename T >
constexpr T squareOf (
    T input ) [constexpr]
```

7.20 DigitalFilters.h

[Go to the documentation of this file.](#)

```

00001 #ifndef _DIGITAL_FILTERS_H_
00002 #define _DIGITAL_FILTERS_H_
00003
00004 #include <stdexcept>
00005 #include <cmath>
00006 #include "CircularDelay.hpp"
00007
00008 template<typename T>
00009 constexpr T squareOf(T input){return input * input;}
00010
00011 namespace tps{
00012     template<typename T>
00013     constexpr T pow(T input, unsigned int power){return (power == 0 ? 1 : input * (power <= 1 ? 1 :
00014         tps::pow(input, power-1)));}
00015 }
00016
00017 template<typename T>
00018 constexpr T calcC_Cr(T k, T m){return 2 * sqrt(k * m);}
00019
00020 template<typename Type>
00021 class DigitalFilter{
00022 public:
00023     virtual Type update(Type newValue) = 0;
00024     virtual Type getOutput() = 0;
00025 };
00026
00027 template<typename T>
00028 class Differentiator : public DigitalFilter<T> {
00029 public:
00030     Differentiator(T sampleTime):
00031         sampleTime(sampleTime)
00032     {}
00033     T update(T input){
00034         y = (input - x1) / sampleTime;
00035         x1 = input;
00036         return y;
00037     }
00038     T getOutput(){return y;}
00039 private:
00040     const T sampleTime;
00041     T y = 0;
00042     T x1 = 0;
00043 };
00044
00045 class LowPassFilter : public DigitalFilter<double> {
00046 public:
00047     LowPassFilter(double idt, double omega_c, double ioutput = 0):
00048         epow(exp(-idt * omega_c)),
00049         dt(idt),
00050         output(ioutput){
00051             if(omega_c < idt){
00052                 throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
00053 sample time dt.");
00054             }
00055         }
00056     double update(double newValue) final{return output = (output-newValue) * epow + newValue;}
00057     double getOutput() final{return output;}
00058     void configOutput(double newOutput){output = newOutput;}
00059     const double* outputPointer(){return &output;}
00060 private:
00061     const double epow;
00062     const double dt;
00063     double output;
00064 };
00065
00066 class LowPassFilter2 : public DigitalFilter<double> {
00067 public:
00068     LowPassFilter2(double dt, double tau_c, double ioutput = 0):
00069         yc{
00070             -2 * (pow(dt, 2) - 4 * pow(tau_c, 2)) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 *
00071 pow(tau_c, 2)),
00072             (-pow(dt, 2) + 2 * sqrt(2) * tau_c * dt - 4 * pow(tau_c, 2)) / (pow(dt, 2) + 2 * sqrt(2) *
00073 tau_c * dt + 4 * pow(tau_c, 2))
00074         },
00075         xc{
00076             pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2)),
00077             2 * pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2)),
00078             pow(dt, 2) / (pow(dt, 2) + 2 * sqrt(2) * tau_c * dt + 4 * pow(tau_c, 2))
00079         }
00080     {
00081         if(tau_c < M_PI * dt){
00082             throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
00083 sample time dt.");
00084         }
00085     }
00086 
```

```

00141         }
00142     }
00150     double update(double newValue) final{
00151         x.push(newValue);
00152         double output = 0;
00153         for (int i = 0; i < 2; ++i)
00154             output += y.get(i) * yc[i];
00155         for (int i = 0; i < 3; ++i)
00156             output += x.get(i) * xc[i];
00157         return y.push(output);
00158     }
00164     double getOutput() final{return y.get(0);}
00173     void configOutput(double newOutput){
00174         for(auto& it : x){
00175             it = newOutput;
00176         }
00177         for(auto& it : y){
00178             it = newOutput;
00179         }
00180     }
00181 private:
00182     const double yc[2];
00183     const double xc[3];
00184     CircularDelay<double, 2> y;
00185     CircularDelay<double, 3> x;
00186 };
00187
00191 class HighPassFilter : public DigitalFilter<double> {
00192 public:
00202     HighPassFilter(double idt, double omega_c):
00203         amplFac(1/((idt * omega_c / 2) + 1)),
00204         ylc((idt * omega_c / 2) - 1),
00205         dt(idt){
00206             if(omega_c < idt){
00207                 throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
00208             }
00209         }
00217     double update(double newValue) final{
00218         // Note that output before assignment equals y1 being y[n-1]
00219         output = amplFac * (newValue - x1 - output * ylc);
00220         x1 = newValue;
00221         return output;
00222     }
00228     double getOutput() final{return output;}
00237     void configOutput(double newOutput){output = newOutput;}
00238     const double* outputPointer() {return &output;}
00239 private:
00240     const double amplFac; // one time calculation constant
00241     const double ylc; // one time calculation constant
00242     const double dt;
00243     double x1 = 0;
00244     double output = 0;
00245 };
00246
00251 class HighPassFilter3 : public DigitalFilter<double> {
00252 public:
00253     HighPassFilter3(double sampleTime, double omega_c, double ioutput = 0):
00254         xc{
00255             8
00256             ,
00257             -24
00258             ,
00259             24
00260             ,
00261             -8
00262         },
00263         yc{
00264             1 / (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c + 8),
00265             3 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) - 8 *
sampleTime * omega_c - 24,
00266             3 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) - 8 *
sampleTime * omega_c + 24,
00267             1 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
sampleTime * omega_c - 8
00268         }
00269     {
00270         if(omega_c < sampleTime){
00271             throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
00272         }
00273     }
00274     double update(double newValue) final{
00275         x.push(newValue);
00276         double y0 = 0;
00277         const double* doubleP = xc;

```

```

00278         for (auto it = x.rbegin(); it != x.rend(); it++)
00279         {
00280             y0 += *it * *doubleP++;
00281         }
00282         doubleP = yc + 1;
00283         for (auto it = y.rbegin(); it != y.rend(); it++)
00284         {
00285             y0 -= *it * *doubleP++;
00286         }
00287         return y.push(yc[0] * y0);
00288         // return y.push(yc[0] * (
00289         //     x.get(0) * xc[0] + x.get(1) * xc[1] + x.get(2) * xc[2] + x.get(3) * xc[3] -
00290         //     y.get(0) * yc[1] - y.get(1) * yc[2] - y.get(2) * yc[3]
00291         // )
00292         // );
00293     }
00294     double getOutput() final{return y.get(0);}
00295 private:
00296     const double xc[4];
00297     const double yc[4];
00298     CircularDelay<double, 3> y;
00299     CircularDelay<double, 4> x;
00300 };
00301
00306 class LowPassFilter3 : public DigitalFilter<double> {
00307 public:
00308     LowPassFilter3(long double sampleTime, long double omega_c, long double ioutput = 0):
00309     {
00310         1
00311         ,
00312         (double)((3 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) -
00313 8 * sampleTime * omega_c - 24)
00314 /
00315 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00316 sampleTime * omega_c + 8))
00317 ,
00318 (double)((3 * tps::pow(sampleTime * omega_c, 3) - 4 * tps::pow(sampleTime * omega_c, 2) -
00319 8 * sampleTime * omega_c + 24)
00320 /
00321 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00322 sampleTime * omega_c + 8))
00323 },
00324 xc{
00325     (double)(1 * tps::pow(sampleTime * omega_c, 3)
00326 /
00327 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00328 sampleTime * omega_c + 8))
00329 ,
00330 (double)(3 * tps::pow(sampleTime * omega_c, 3)
00331 /
00332 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00333 sampleTime * omega_c + 8))
00334 ,
00335 (double)(3 * tps::pow(sampleTime * omega_c, 3)
00336 /
00337 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00338 sampleTime * omega_c + 8))
00339 ,
00340 (double)(1 * tps::pow(sampleTime * omega_c, 3)
00341 /
00342 (1 * tps::pow(sampleTime * omega_c, 3) + 4 * tps::pow(sampleTime * omega_c, 2) + 8 *
00343 sampleTime * omega_c + 8))
00344 }
00345 {
00346     if(omega_c < sampleTime){
00347         throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
00348 sample time dt.");
00349     }
00350 }
00351 double update(double newValue) final{
00352     x.push(newValue);
00353     double y0 = 0;
00354     const double* doubleP = xc;
00355     for (auto it = x.rbegin(); it != x.rend(); it++)
00356     {
00357         y0 += *it * *doubleP++;
00358     }
00359     doubleP = yc + 1;
00360     for (auto it = y.rbegin(); it != y.rend(); it++)
00361     {
00362         y0 -= *it * *doubleP++;
00363     }

```

```

00358     }
00359     return y.push(y0);
00360 }
00361 double getOutput() final{return y.get(0);}
00362 private:
00363     const double yc[4];
00364     const double xc[4];
00365     CircularDelay<double, 3> y;
00366     CircularDelay<double, 4> x;
00367 };
00368
00373 class LowPassFilter3MatchedZ : public DigitalFilter<double> {
00374 public:
00375     LowPassFilter3MatchedZ(long double sampleTime, long double omega_c):
00376         amplFac(-(2*(exp(3 * omega_c * sampleTime) - exp(2 * omega_c * sampleTime))*cosl(sqrtl(3) *
omega_c * sampleTime / 2) - exp(7 * omega_c * sampleTime / 2) + exp(3 * omega_c * sampleTime /
2))*exp(-7 * omega_c * sampleTime / 2)),
00377         yc{
00378             (double)(-(2 * cosl(sqrtl(3) * omega_c * sampleTime / 2) * exp(omega_c * sampleTime * 5 /
2) + exp(2 * omega_c * sampleTime)) * exp(-3 * omega_c * sampleTime))
00379             ,
00380             (double)((2 * cosl(sqrtl(3) * omega_c * sampleTime / 2) * exp(omega_c * sampleTime * 3 /
2) + exp(2 * omega_c * sampleTime)) * exp(-3 * omega_c * sampleTime))
00381             ,
00382             (double)(-exp(-2 * omega_c * sampleTime))
00383         }
00384     {
00385         if(omega_c / (2 * M_PI) < sampleTime){
00386             throw std::domain_error("LowPassFilter3MatchedZ constructor error: tua_c is smaller
than the sample time dt.");
00387         }
00388     }
00389     double update(double newValue) final{
00390         double y0 = newValue * amplFac;
00391         const double* doubleP = yc;
00392         for (auto it = y.rbegin(); it != y.rend(); it++)
00393         {
00394             y0 -= *it * *doubleP++;
00395         }
00396         return y.push(y0);
00397     }
00398     double getOutput() final{return y.get(0);}
00399 private:
00400     const double amplFac;
00401     const double yc[3];
00402     CircularDelay<double, 3> y;
00403 };
00404
00409 class LowPassFilter3DiffApprox : public DigitalFilter<double> {
00410 public:
00411     LowPassFilter3DiffApprox(double sampleTime, double omega_c, double ioutput = 0):
00412         xc{
00413             1 * tps::pow(sampleTime * omega_c, 3)
00414             ,
00415             0
00416             ,
00417             0
00418             ,
00419             0
00420         },
00421         yc{
00422             1 / (1 * tps::pow(sampleTime * omega_c, 3) + 2 * tps::pow(sampleTime * omega_c, 2) + 2 *
sampleTime * omega_c + 1),
00423             0 * tps::pow(sampleTime * omega_c, 3) - 2 * tps::pow(sampleTime * omega_c, 2) - 4 *
sampleTime * omega_c - 3,
00424             0 * tps::pow(sampleTime * omega_c, 3) + 0 * tps::pow(sampleTime * omega_c, 2) + 2 *
sampleTime * omega_c + 3,
00425             0 * tps::pow(sampleTime * omega_c, 3) - 0 * tps::pow(sampleTime * omega_c, 2) + 0 *
sampleTime * omega_c - 1
00426         }
00427     {
00428         if(omega_c < sampleTime){
00429             throw std::domain_error("LowPassFilter constructor error: tua_c is smaller than the
sample time dt.");
00430         }
00431     }
00432     double update(double newValue) final{
00433         x.push(newValue);
00434         double y0 = 0;
00435         const double* doubleP = xc;
00436         for (auto it = x.rbegin(); it != x.rend(); it++)
00437         {
00438             y0 += *it * *doubleP++;
00439         }
00440         doubleP = yc + 1;
00441         for (auto it = y.rbegin(); it != y.rend(); it++)
00442         {

```

```

00443         y0 -= *it * *doubleP++;
00444     }
00445     return y.push(y0 * y0);
00446 }
00447 double getOutput() final{return y.get(0);}
00448 private:
00449     const double xc[4];
00450     const double yc[4];
00451     CircularDelay<double, 3> y;
00452     CircularDelay<double, 4> x;
00453 };
00454
00455 template<size_t size>
00456 class MovingAvarageFilter{
00457 public:
00458     double update(double input){
00459         input *= 1000;
00460         output += int64_t(input) - *buffer.rbegin();
00461         buffer.push(input);
00462         return double(output) / (1000);
00463     }
00464 private:
00465     int64_t output = 0;
00466     CircularDelay<int64_t, size> buffer;
00467 };
00468
00469 #endif

```

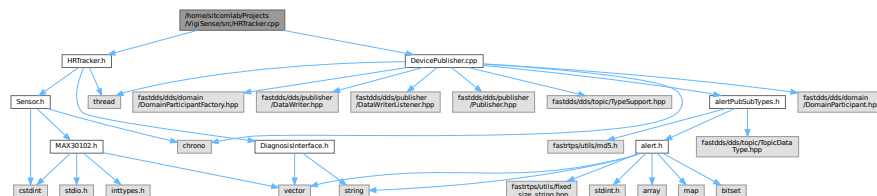
7.21 /home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp File Reference

```

#include "HRTracker.h"
#include "DevicePublisher.cpp"

```

Include dependency graph for HRTracker.cpp:



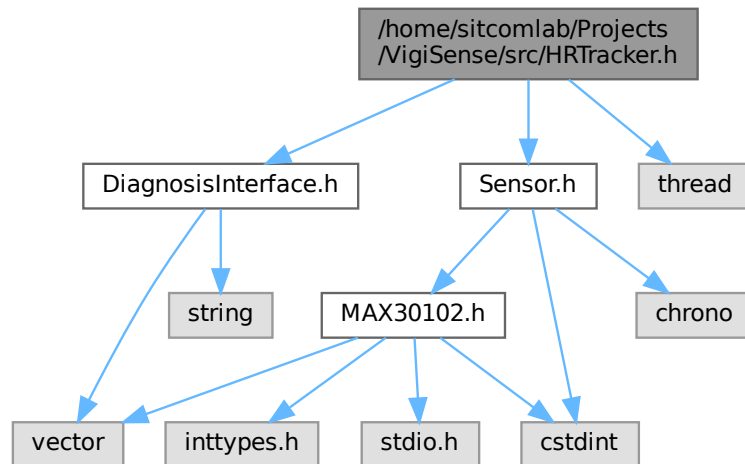
7.22 /home/sitcomlab/Projects/VigiSense/src/HRTracker.h File Reference

```

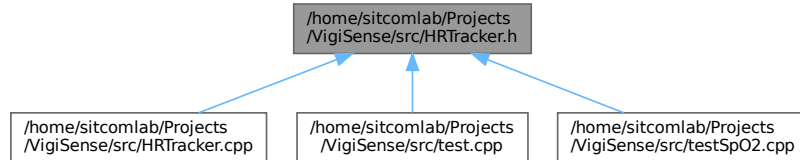
#include "DiagnosisInterface.h"
#include "Sensor.h"
#include <thread>

```

Include dependency graph for HRTracker.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [HRTracker](#)

7.23 HRTracker.h

[Go to the documentation of this file.](#)

```

00001 #pragma once
00002 #include "DiagnosisInterface.h"
00003 #include "Sensor.h"
00004 #include <thread>
00005
00006 class HRTracker:public diagnosisInterface {
00007     public:
00008         HRTracker(sensor *s);
00009         ~HRTracker();
00010         void start();
00011         void stop();
00012         void ping();
00013         int getVal();
00014         void tracker();
  
```



```

00015     protected:
00016         sensor* _s;
00017         bool threadRunning = false;
00018         void pingThread();
00019         // define symptom table here
00020         std::vector<symptomRange> symptomRanges {
00021             {0,60,"Bradycardia"},
00022             {60,100,"Normal resting heart rate"},
00023             {100,200,"Tachycardia"}};
00024
00025 };

```

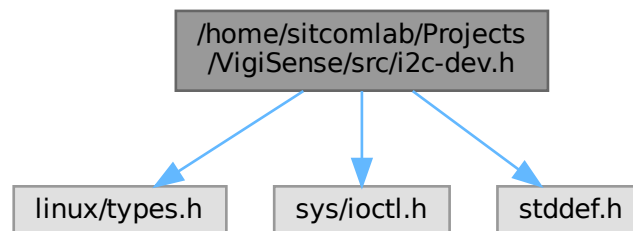
7.24 /home/sitcomlab/Projects/VigiSense/src/i2c-dev.h File Reference

```

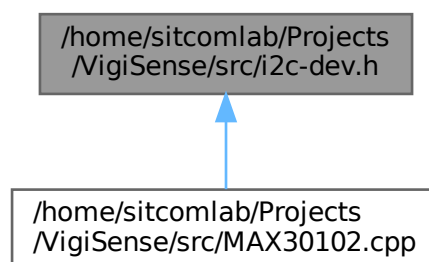
#include <linux/types.h>
#include <sys/ioctl.h>
#include <stddef.h>

```

Include dependency graph for i2c-dev.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct [i2c_msg](#)
- union [i2c_smbus_data](#)
- struct [i2c_smbus_ioctl_data](#)
- struct [i2c_rdwr_ioctl_data](#)

Macros

- `#define I2C_M_TEN 0x10 /* we have a ten bit chip address */`
- `#define I2C_M_RD 0x01`
- `#define I2C_M_NOSTART 0x4000`
- `#define I2C_M_REV_DIR_ADDR 0x2000`
- `#define I2C_M_IGNORE_NAK 0x1000`
- `#define I2C_M_NO_RD_ACK 0x0800`
- `#define I2C_FUNC_I2C 0x00000001`
- `#define I2C_FUNC_10BIT_ADDR 0x00000002`
- `#define I2C_FUNC_PROTOCOL_MANGLING 0x00000004 /* I2C_M_{REV_DIR_ADDR,NOSTART,..} */`
- `#define I2C_FUNC_SMBUS_PEC 0x00000008`
- `#define I2C_FUNC_SMBUS_BLOCK_PROC_CALL 0x00008000 /* SMBus 2.0 */`
- `#define I2C_FUNC_SMBUS_QUICK 0x00010000`
- `#define I2C_FUNC_SMBUS_READ_BYTE 0x00020000`
- `#define I2C_FUNC_SMBUS_WRITE_BYTE 0x00040000`
- `#define I2C_FUNC_SMBUS_READ_BYTE_DATA 0x00080000`
- `#define I2C_FUNC_SMBUS_WRITE_BYTE_DATA 0x00100000`
- `#define I2C_FUNC_SMBUS_READ_WORD_DATA 0x00200000`
- `#define I2C_FUNC_SMBUS_WRITE_WORD_DATA 0x00400000`
- `#define I2C_FUNC_SMBUS_PROC_CALL 0x00800000`
- `#define I2C_FUNC_SMBUS_READ_BLOCK_DATA 0x01000000`
- `#define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000`
- `#define I2C_FUNC_SMBUS_READ_I2C_BLOCK 0x04000000 /* I2C-like block xfer */`
- `#define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK 0x08000000 /* w/ 1-byte reg. addr. */`
- `#define I2C_FUNC_SMBUS_BYTE`
- `#define I2C_FUNC_SMBUS_BYTE_DATA`
- `#define I2C_FUNC_SMBUS_WORD_DATA`
- `#define I2C_FUNC_SMBUS_BLOCK_DATA`
- `#define I2C_FUNC_SMBUS_I2C_BLOCK`
- `#define I2C_FUNC_SMBUS_HWPEC_CALC I2C_FUNC_SMBUS_PEC`
- `#define I2C_SMBUS_BLOCK_MAX 32 /* As specified in SMBus standard */`
- `#define I2C_SMBUS_I2C_BLOCK_MAX 32 /* Not specified but we use same structure */`
- `#define I2C_SMBUS_READ 1`
- `#define I2C_SMBUS_WRITE 0`
- `#define I2C_SMBUS_QUICK 0`
- `#define I2C_SMBUS_BYTE 1`
- `#define I2C_SMBUS_BYTE_DATA 2`
- `#define I2C_SMBUS_WORD_DATA 3`
- `#define I2C_SMBUS_PROC_CALL 4`
- `#define I2C_SMBUS_BLOCK_DATA 5`
- `#define I2C_SMBUS_I2C_BLOCK_BROKEN 6`
- `#define I2C_SMBUS_BLOCK_PROC_CALL 7 /* SMBus 2.0 */`
- `#define I2C_SMBUS_I2C_BLOCK_DATA 8`
- `#define I2C_RETRIES`
- `#define I2C_TIMEOUT 0x0702 /* set timeout in units of 10 ms */`
- `#define I2C_SLAVE 0x0703 /* Use this slave address */`
- `#define I2C_SLAVE_FORCE`
- `#define I2C_TENBIT 0x0704 /* 0 for 7 bit addrs, != 0 for 10 bit */`
- `#define I2C_FUNCS 0x0705 /* Get the adapter functionality mask */`
- `#define I2C_RDWR 0x0707 /* Combined R/W transfer (one STOP only) */`
- `#define I2C_PEC 0x0708 /* != 0 to use PEC with SMBus */`
- `#define I2C_SMBUS 0x0720 /* SMBus transfer */`
- `#define I2C_RDRW_IOCTL_MAX_MSGS 42`

Functions

- static __s32 [i2c_smbus_access](#) (int file, char read_write, __u8 command, int size, union [i2c_smbus_data](#) *data)
- static __s32 [i2c_smbus_write_quick](#) (int file, __u8 value)
- static __s32 [i2c_smbus_read_byte](#) (int file)
- static __s32 [i2c_smbus_write_byte](#) (int file, __u8 value)
- static __s32 [i2c_smbus_read_byte_data](#) (int file, __u8 command)
- static __s32 [i2c_smbus_write_byte_data](#) (int file, __u8 command, __u8 value)
- static __s32 [i2c_smbus_read_word_data](#) (int file, __u8 command)
- static __s32 [i2c_smbus_write_word_data](#) (int file, __u8 command, __u16 value)
- static __s32 [i2c_smbus_process_call](#) (int file, __u8 command, __u16 value)
- static __s32 [i2c_smbus_read_block_data](#) (int file, __u8 command, __u8 *values)
- static __s32 [i2c_smbus_write_block_data](#) (int file, __u8 command, __u8 length, const __u8 *values)
- static __s32 [i2c_smbus_read_i2c_block_data](#) (int file, __u8 command, __u8 length, __u8 *values)
- static __s32 [i2c_smbus_write_i2c_block_data](#) (int file, __u8 command, __u8 length, const __u8 *values)
- static __s32 [i2c_smbus_block_process_call](#) (int file, __u8 command, __u8 length, __u8 *values)

7.24.1 Macro Definition Documentation

7.24.1.1 I2C_FUNC_10BIT_ADDR

```
#define I2C_FUNC_10BIT_ADDR 0x00000002
```

7.24.1.2 I2C_FUNC_I2C

```
#define I2C_FUNC_I2C 0x00000001
```

7.24.1.3 I2C_FUNC_PROTOCOL_MANGLING

```
#define I2C_FUNC_PROTOCOL_MANGLING 0x00000004 /* I2C_M_{REV_DIR_ADDR,NOSTART,...} */
```

7.24.1.4 I2C_FUNC_SMBUS_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_BLOCK_DATA
```

Value:

```
(I2C_FUNC_SMBUS_READ_BLOCK_DATA | \
 I2C_FUNC_SMBUS_WRITE_BLOCK_DATA)
```

7.24.1.5 I2C_FUNC_SMBUS_BLOCK_PROC_CALL

```
#define I2C_FUNC_SMBUS_BLOCK_PROC_CALL 0x00008000 /* SMBus 2.0 */
```

7.24.1.6 I2C_FUNC_SMBUS_BYTE

```
#define I2C_FUNC_SMBUS_BYTE
```

Value:

```
(I2C_FUNC_SMBUS_READ_BYTE | \  
I2C_FUNC_SMBUS_WRITE_BYTE)
```

7.24.1.7 I2C_FUNC_SMBUS_BYTE_DATA

```
#define I2C_FUNC_SMBUS_BYTE_DATA
```

Value:

```
(I2C_FUNC_SMBUS_READ_BYTE_DATA | \  
I2C_FUNC_SMBUS_WRITE_BYTE_DATA)
```

7.24.1.8 I2C_FUNC_SMBUS_HWPEC_CALC

```
#define I2C_FUNC_SMBUS_HWPEC_CALC I2C_FUNC_SMBUS_PEC
```

7.24.1.9 I2C_FUNC_SMBUS_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_I2C_BLOCK
```

Value:

```
(I2C_FUNC_SMBUS_READ_I2C_BLOCK | \  
I2C_FUNC_SMBUS_WRITE_I2C_BLOCK)
```

7.24.1.10 I2C_FUNC_SMBUS_PEC

```
#define I2C_FUNC_SMBUS_PEC 0x00000008
```

7.24.1.11 I2C_FUNC_SMBUS_PROC_CALL

```
#define I2C_FUNC_SMBUS_PROC_CALL 0x00800000
```

7.24.1.12 I2C_FUNC_SMBUS_QUICK

```
#define I2C_FUNC_SMBUS_QUICK 0x00010000
```

7.24.1.13 I2C_FUNC_SMBUS_READ_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_READ_BLOCK_DATA 0x01000000
```

7.24.1.14 I2C_FUNC_SMBUS_READ_BYTE

```
#define I2C_FUNC_SMBUS_READ_BYTE 0x00020000
```

7.24.1.15 I2C_FUNC_SMBUS_READ_BYTE_DATA

```
#define I2C_FUNC_SMBUS_READ_BYTE_DATA 0x00080000
```

7.24.1.16 I2C_FUNC_SMBUS_READ_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_READ_I2C_BLOCK 0x04000000 /* I2C-like block xfer */
```

7.24.1.17 I2C_FUNC_SMBUS_READ_WORD_DATA

```
#define I2C_FUNC_SMBUS_READ_WORD_DATA 0x00200000
```

7.24.1.18 I2C_FUNC_SMBUS_WORD_DATA

```
#define I2C_FUNC_SMBUS_WORD_DATA
```

Value:

```
(I2C_FUNC_SMBUS_READ_WORD_DATA | \  
I2C_FUNC_SMBUS_WRITE_WORD_DATA)
```

7.24.1.19 I2C_FUNC_SMBUS_WRITE_BLOCK_DATA

```
#define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000
```

7.24.1.20 I2C_FUNC_SMBUS_WRITE_BYTE

```
#define I2C_FUNC_SMBUS_WRITE_BYTE 0x00040000
```

7.24.1.21 I2C_FUNC_SMBUS_WRITE_BYTE_DATA

```
#define I2C_FUNC_SMBUS_WRITE_BYTE_DATA 0x00100000
```

7.24.1.22 I2C_FUNC_SMBUS_WRITE_I2C_BLOCK

```
#define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK 0x08000000 /* w/ 1-byte reg. addr. */
```

7.24.1.23 I2C_FUNC_SMBUS_WRITE_WORD_DATA

```
#define I2C_FUNC_SMBUS_WRITE_WORD_DATA 0x00400000
```

7.24.1.24 I2C_FUNCS

```
#define I2C_FUNCS 0x0705 /* Get the adapter functionality mask */
```

7.24.1.25 I2C_M_IGNORE_NAK

```
#define I2C_M_IGNORE_NAK 0x1000
```

7.24.1.26 I2C_M_NO_RD_ACK

```
#define I2C_M_NO_RD_ACK 0x0800
```

7.24.1.27 I2C_M_NOSTART

```
#define I2C_M_NOSTART 0x4000
```

7.24.1.28 I2C_M_RD

```
#define I2C_M_RD 0x01
```

7.24.1.29 I2C_M_REV_DIR_ADDR

```
#define I2C_M_REV_DIR_ADDR 0x2000
```

7.24.1.30 I2C_M_TEN

```
#define I2C_M_TEN 0x10 /* we have a ten bit chip address */
```

7.24.1.31 I2C_PEC

```
#define I2C_PEC 0x0708 /* != 0 to use PEC with SMBus */
```

7.24.1.32 I2C_RDRW_IOCTL_MAX_MSGS

```
#define I2C_RDRW_IOCTL_MAX_MSGS 42
```

7.24.1.33 I2C_RDWR

```
#define I2C_RDWR 0x0707 /* Combined R/W transfer (one STOP only) */
```

7.24.1.34 I2C_RETRIES

```
#define I2C_RETRIES
```

Value:

```
0x0701 /* number of times a device address should  
be polled when not acknowledging */
```

7.24.1.35 I2C_SLAVE

```
#define I2C_SLAVE 0x0703 /* Use this slave address */
```

7.24.1.36 I2C_SLAVE_FORCE

```
#define I2C_SLAVE_FORCE
```

Value:

```
0x0706 /* Use this slave address, even if it  
is already in use by a driver! */
```

7.24.1.37 I2C_SMBUS

```
#define I2C_SMBUS 0x0720 /* SMBus transfer */
```

7.24.1.38 I2C_SMBUS_BLOCK_DATA

```
#define I2C_SMBUS_BLOCK_DATA 5
```

7.24.1.39 I2C_SMBUS_BLOCK_MAX

```
#define I2C_SMBUS_BLOCK_MAX 32 /* As specified in SMBus standard */
```

7.24.1.40 I2C_SMBUS_BLOCK_PROC_CALL

```
#define I2C_SMBUS_BLOCK_PROC_CALL 7 /* SMBus 2.0 */
```

7.24.1.41 I2C_SMBUS_BYTE

```
#define I2C_SMBUS_BYTE 1
```

7.24.1.42 I2C_SMBUS_BYTE_DATA

```
#define I2C_SMBUS_BYTE_DATA 2
```

7.24.1.43 I2C_SMBUS_I2C_BLOCK_BROKEN

```
#define I2C_SMBUS_I2C_BLOCK_BROKEN 6
```

7.24.1.44 I2C_SMBUS_I2C_BLOCK_DATA

```
#define I2C_SMBUS_I2C_BLOCK_DATA 8
```

7.24.1.45 I2C_SMBUS_I2C_BLOCK_MAX

```
#define I2C_SMBUS_I2C_BLOCK_MAX 32 /* Not specified but we use same structure */
```

7.24.1.46 I2C_SMBUS_PROC_CALL

```
#define I2C_SMBUS_PROC_CALL 4
```

7.24.1.47 I2C_SMBUS_QUICK

```
#define I2C_SMBUS_QUICK 0
```

7.24.1.48 I2C_SMBUS_READ

```
#define I2C_SMBUS_READ 1
```

7.24.1.49 I2C_SMBUS_WORD_DATA

```
#define I2C_SMBUS_WORD_DATA 3
```

7.24.1.50 I2C_SMBUS_WRITE

```
#define I2C_SMBUS_WRITE 0
```

7.24.1.51 I2C_TENBIT

```
#define I2C_TENBIT 0x0704 /* 0 for 7 bit addrs, != 0 for 10 bit */
```

7.24.1.52 I2C_TIMEOUT

```
#define I2C_TIMEOUT 0x0702 /* set timeout in units of 10 ms */
```


7.24.2 Function Documentation

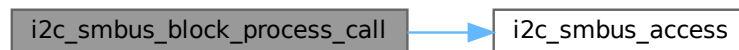
7.24.2.1 i2c_smbus_access()

```
static __s32 i2c_smbus_access (
    int file,
    char read_write,
    __u8 command,
    int size,
    union i2c_smbus_data * data ) [inline], [static]
```

7.24.2.2 i2c_smbus_block_process_call()

```
static __s32 i2c_smbus_block_process_call (
    int file,
    __u8 command,
    __u8 length,
    __u8 * values ) [inline], [static]
```

Here is the call graph for this function:



7.24.2.3 i2c_smbus_process_call()

```
static __s32 i2c_smbus_process_call (
    int file,
    __u8 command,
    __u16 value ) [inline], [static]
```

7.24.2.4 i2c_smbus_read_block_data()

```
static __s32 i2c_smbus_read_block_data (
    int file,
    __u8 command,
    __u8 * values ) [inline], [static]
```

Here is the call graph for this function:



7.24.2.5 i2c_smbus_read_byte()

```
static __s32 i2c_smbus_read_byte (  
    int file )    [inline], [static]
```

7.24.2.6 i2c_smbus_read_byte_data()

```
static __s32 i2c_smbus_read_byte_data (  
    int file,  
    __u8 command )    [inline], [static]
```

7.24.2.7 i2c_smbus_read_i2c_block_data()

```
static __s32 i2c_smbus_read_i2c_block_data (  
    int file,  
    __u8 command,  
    __u8 length,  
    __u8 * values )    [inline], [static]
```

Here is the call graph for this function:



7.24.2.8 i2c_smbus_read_word_data()

```
static __s32 i2c_smbus_read_word_data (  
    int file,  
    __u8 command )    [inline], [static]
```

7.24.2.9 i2c_smbus_write_block_data()

```
static __s32 i2c_smbus_write_block_data (  
    int file,  
    __u8 command,  
    __u8 length,  
    const __u8 * values )    [inline], [static]
```

7.24.2.10 i2c_smbus_write_byte()

```
static __s32 i2c_smbus_write_byte (
    int file,
    __u8 value ) [inline], [static]
```

7.24.2.11 i2c_smbus_write_byte_data()

```
static __s32 i2c_smbus_write_byte_data (
    int file,
    __u8 command,
    __u8 value ) [inline], [static]
```

7.24.2.12 i2c_smbus_write_i2c_block_data()

```
static __s32 i2c_smbus_write_i2c_block_data (
    int file,
    __u8 command,
    __u8 length,
    const __u8 * values ) [inline], [static]
```

7.24.2.13 i2c_smbus_write_quick()

```
static __s32 i2c_smbus_write_quick (
    int file,
    __u8 value ) [inline], [static]
```

Here is the call graph for this function:



7.24.2.14 i2c_smbus_write_word_data()

```
static __s32 i2c_smbus_write_word_data (
    int file,
    __u8 command,
    __u16 value ) [inline], [static]
```

7.25 i2c-dev.h

[Go to the documentation of this file.](#)

```

00001  /*
00002      i2c-dev.h - i2c-bus driver, char device interface
00003
00004      Copyright (C) 1995-97 Simon G. Vogl
00005      Copyright (C) 1998-99 Frodo Looijaard <frodol@dds.nl>
00006
00007      This program is free software; you can redistribute it and/or modify
00008      it under the terms of the GNU General Public License as published by
00009      the Free Software Foundation; either version 2 of the License, or
00010      (at your option) any later version.
00011
00012      This program is distributed in the hope that it will be useful,
00013      but WITHOUT ANY WARRANTY; without even the implied warranty of
00014      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
00015      GNU General Public License for more details.
00016
00017      You should have received a copy of the GNU General Public License
00018      along with this program; if not, write to the Free Software
00019      Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston,
00020      MA 02110-1301 USA.
00021  */
00022
00023  #ifndef _LINUX_I2C_DEV_H
00024  #define _LINUX_I2C_DEV_H
00025
00026  #include <linux/types.h>
00027  #include <sys/ioctl.h>
00028  #include <stddef.h>
00029
00030
00031  /* -- i2c.h -- */
00032
00033
00034  /*
00035   * I2C Message - used for pure i2c transaction, also from /dev interface
00036   */
00037  struct i2c_msg {
00038      __u16 addr; /* slave address */
00039      unsigned short flags;
00040      #define I2C_M_TEN 0x10 /* we have a ten bit chip address */
00041      #define I2C_M_RD 0x01
00042      #define I2C_M_NOSTART 0x4000
00043      #define I2C_M_REV_DIR_ADDR 0x2000
00044      #define I2C_M_IGNORE_NAK 0x1000
00045      #define I2C_M_NO_RD_ACK 0x0800
00046      short len; /* msg length */
00047      char *buf; /* pointer to msg data */
00048  };
00049
00050  /* To determine what functionality is present */
00051
00052  #define I2C_FUNC_I2C 0x00000001
00053  #define I2C_FUNC_10BIT_ADDR 0x00000002
00054  #define I2C_FUNC_PROTOCOL_MANGLING 0x00000004 /* I2C_M_{REV_DIR_ADDR,NOSTART,..} */
00055  #define I2C_FUNC_SMBUS_PEC 0x00000008
00056  #define I2C_FUNC_SMBUS_BLOCK_PROC_CALL 0x00008000 /* SMBus 2.0 */
00057  #define I2C_FUNC_SMBUS_QUICK 0x00010000
00058  #define I2C_FUNC_SMBUS_READ_BYTE 0x00020000
00059  #define I2C_FUNC_SMBUS_WRITE_BYTE 0x00040000
00060  #define I2C_FUNC_SMBUS_READ_BYTE_DATA 0x00080000
00061  #define I2C_FUNC_SMBUS_WRITE_BYTE_DATA 0x00100000
00062  #define I2C_FUNC_SMBUS_READ_WORD_DATA 0x00200000
00063  #define I2C_FUNC_SMBUS_WRITE_WORD_DATA 0x00400000
00064  #define I2C_FUNC_SMBUS_PROC_CALL 0x00800000
00065  #define I2C_FUNC_SMBUS_READ_BLOCK_DATA 0x01000000
00066  #define I2C_FUNC_SMBUS_WRITE_BLOCK_DATA 0x02000000
00067  #define I2C_FUNC_SMBUS_READ_I2C_BLOCK 0x04000000 /* I2C-like block xfer */
00068  #define I2C_FUNC_SMBUS_WRITE_I2C_BLOCK 0x08000000 /* w/ 1-byte reg. addr. */
00069
00070  #define I2C_FUNC_SMBUS_BYTE (I2C_FUNC_SMBUS_READ_BYTE | \
00071                               I2C_FUNC_SMBUS_WRITE_BYTE)
00072  #define I2C_FUNC_SMBUS_BYTE_DATA (I2C_FUNC_SMBUS_READ_BYTE_DATA | \
00073                                     I2C_FUNC_SMBUS_WRITE_BYTE_DATA)
00074  #define I2C_FUNC_SMBUS_WORD_DATA (I2C_FUNC_SMBUS_READ_WORD_DATA | \
00075                                     I2C_FUNC_SMBUS_WRITE_WORD_DATA)
00076  #define I2C_FUNC_SMBUS_BLOCK_DATA (I2C_FUNC_SMBUS_READ_BLOCK_DATA | \
00077                                       I2C_FUNC_SMBUS_WRITE_BLOCK_DATA)
00078  #define I2C_FUNC_SMBUS_I2C_BLOCK (I2C_FUNC_SMBUS_READ_I2C_BLOCK | \
00079                                       I2C_FUNC_SMBUS_WRITE_I2C_BLOCK)
00080
00081  /* Old name, for compatibility */
00082  #define I2C_FUNC_SMBUS_HWPEC_CALC I2C_FUNC_SMBUS_PEC

```

```

00083
00084 /*
00085  * Data for SMBus Messages
00086  */
00087 #define I2C_SMBUS_BLOCK_MAX      32      /* As specified in SMBus standard */
00088 #define I2C_SMBUS_I2C_BLOCK_MAX  32      /* Not specified but we use same structure */
00089 union i2c_smbus_data {
00090     __u8 byte;
00091     __u16 word;
00092     __u8 block[I2C_SMBUS_BLOCK_MAX + 2]; /* block[0] is used for length */
00093                                           /* and one more for PEC */
00094 };
00095
00096 /* smbus_access read or write markers */
00097 #define I2C_SMBUS_READ  1
00098 #define I2C_SMBUS_WRITE 0
00099
00100 /* SMBus transaction types (size parameter in the above functions)
00101  Note: these no longer correspond to the (arbitrary) PIIX4 internal codes! */
00102 #define I2C_SMBUS_QUICK      0
00103 #define I2C_SMBUS_BYTE      1
00104 #define I2C_SMBUS_BYTE_DATA  2
00105 #define I2C_SMBUS_WORD_DATA  3
00106 #define I2C_SMBUS_PROC_CALL  4
00107 #define I2C_SMBUS_BLOCK_DATA 5
00108 #define I2C_SMBUS_I2C_BLOCK_BROKEN 6
00109 #define I2C_SMBUS_BLOCK_PROC_CALL 7 /* SMBus 2.0 */
00110 #define I2C_SMBUS_I2C_BLOCK_DATA 8
00111
00112
00113 /* /dev/i2c-X ioctl commands. The ioctl's parameter is always an
00114  * unsigned long, except for:
00115  *   - I2C_FUNCS, takes pointer to an unsigned long
00116  *   - I2C_RDWR, takes pointer to struct i2c_rdwr_ioctl_data
00117  *   - I2C_SMBUS, takes pointer to struct i2c_smbus_ioctl_data
00118  */
00119 #define I2C_RETRIES      0x0701 /* number of times a device address should
00120                                  be polled when not acknowledging */
00121 #define I2C_TIMEOUT      0x0702 /* set timeout in units of 10 ms */
00122
00123 /* NOTE: Slave address is 7 or 10 bits, but 10-bit addresses
00124  * are NOT supported! (due to code brokenness)
00125  */
00126 #define I2C_SLAVE      0x0703 /* Use this slave address */
00127 #define I2C_SLAVE_FORCE 0x0706 /* Use this slave address, even if it
00128                                  is already in use by a driver! */
00129 #define I2C_TENBIT      0x0704 /* 0 for 7 bit addrs, != 0 for 10 bit */
00130
00131 #define I2C_FUNCS      0x0705 /* Get the adapter functionality mask */
00132
00133 #define I2C_RDWR      0x0707 /* Combined R/W transfer (one STOP only) */
00134
00135 #define I2C_PEC      0x0708 /* != 0 to use PEC with SMBus */
00136 #define I2C_SMBUS      0x0720 /* SMBus transfer */
00137
00138
00139 /* This is the structure as used in the I2C_SMBUS ioctl call */
00140 struct i2c_smbus_ioctl_data {
00141     __u8 read_write;
00142     __u8 command;
00143     __u32 size;
00144     union i2c_smbus_data *data;
00145 };
00146
00147 /* This is the structure as used in the I2C_RDWR ioctl call */
00148 struct i2c_rdwr_ioctl_data {
00149     struct i2c_msg *msgs; /* pointers to i2c_msgs */
00150     __u32 nmsgs; /* number of i2c_msgs */
00151 };
00152
00153 #define I2C_RDRW_IOCTL_MAX_MSGS      42
00154
00155
00156 static inline __s32 i2c_smbus_access(int file, char read_write, __u8 command,
00157                                     int size, union i2c_smbus_data *data)
00158 {
00159     struct i2c_smbus_ioctl_data args;
00160
00161     args.read_write = read_write;
00162     args.command = command;
00163     args.size = size;
00164     args.data = data;
00165     return ioctl(file, I2C_SMBUS, &args);
00166 }
00167
00168
00169 static inline __s32 i2c_smbus_write_quick(int file, __u8 value)

```

```

00170 {
00171     return i2c_smbus_access(file, value, 0, I2C_SMBUS_QUICK, NULL);
00172 }
00173
00174 static inline __s32 i2c_smbus_read_byte(int file)
00175 {
00176     union i2c_smbus_data data;
00177     if (i2c_smbus_access(file, I2C_SMBUS_READ, 0, I2C_SMBUS_BYTE, &data))
00178         return -1;
00179     else
00180         return 0xFF & data.byte;
00181 }
00182
00183 static inline __s32 i2c_smbus_write_byte(int file, __u8 value)
00184 {
00185     return i2c_smbus_access(file, I2C_SMBUS_WRITE, value,
00186                             I2C_SMBUS_BYTE, NULL);
00187 }
00188
00189 static inline __s32 i2c_smbus_read_byte_data(int file, __u8 command)
00190 {
00191     union i2c_smbus_data data;
00192     if (i2c_smbus_access(file, I2C_SMBUS_READ, command,
00193                           I2C_SMBUS_BYTE_DATA, &data))
00194         return -1;
00195     else
00196         return 0xFF & data.byte;
00197 }
00198
00199 static inline __s32 i2c_smbus_write_byte_data(int file, __u8 command,
00200                                               __u8 value)
00201 {
00202     union i2c_smbus_data data;
00203     data.byte = value;
00204     return i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00205                             I2C_SMBUS_BYTE_DATA, &data);
00206 }
00207
00208 static inline __s32 i2c_smbus_read_word_data(int file, __u8 command)
00209 {
00210     union i2c_smbus_data data;
00211     if (i2c_smbus_access(file, I2C_SMBUS_READ, command,
00212                           I2C_SMBUS_WORD_DATA, &data))
00213         return -1;
00214     else
00215         return 0xFFFF & data.word;
00216 }
00217
00218 static inline __s32 i2c_smbus_write_word_data(int file, __u8 command,
00219                                               __u16 value)
00220 {
00221     union i2c_smbus_data data;
00222     data.word = value;
00223     return i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00224                             I2C_SMBUS_WORD_DATA, &data);
00225 }
00226
00227 static inline __s32 i2c_smbus_process_call(int file, __u8 command, __u16 value)
00228 {
00229     union i2c_smbus_data data;
00230     data.word = value;
00231     if (i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00232                           I2C_SMBUS_PROC_CALL, &data))
00233         return -1;
00234     else
00235         return 0xFFFF & data.word;
00236 }
00237
00238 /* Returns the number of read bytes */
00239 static inline __s32 i2c_smbus_read_block_data(int file, __u8 command,
00240                                               __u8 *values)
00241 {
00242     union i2c_smbus_data data;
00243     int i;
00244     if (i2c_smbus_access(file, I2C_SMBUS_READ, command,
00245                           I2C_SMBUS_BLOCK_DATA, &data))
00246         return -1;
00247     else {
00248         for (i = 1; i <= data.block[0]; i++)
00249             values[i-1] = data.block[i];
00250         return data.block[0];
00251     }
00252 }
00253
00254 static inline __s32 i2c_smbus_write_block_data(int file, __u8 command,
00255                                               __u8 length, const __u8 *values)

```

```

00257 {
00258     union i2c_smbus_data data;
00259     int i;
00260     if (length > 32)
00261         length = 32;
00262     for (i = 1; i <= length; i++)
00263         data.block[i] = values[i-1];
00264     data.block[0] = length;
00265     return i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00266                             I2C_SMBUS_BLOCK_DATA, &data);
00267 }
00268
00269 /* Returns the number of read bytes */
00270 /* Until kernel 2.6.22, the length is hardcoded to 32 bytes. If you
00271    ask for less than 32 bytes, your code will only work with kernels
00272    2.6.23 and later. */
00273 static inline __s32 i2c_smbus_read_i2c_block_data(int file, __u8 command,
00274                                                  __u8 length, __u8 *values)
00275 {
00276     union i2c_smbus_data data;
00277     int i;
00278
00279     if (length > 32)
00280         length = 32;
00281     data.block[0] = length;
00282     if (i2c_smbus_access(file, I2C_SMBUS_READ, command,
00283                          length == 32 ? I2C_SMBUS_I2C_BLOCK_BROKEN :
00284                          I2C_SMBUS_I2C_BLOCK_DATA, &data))
00285         return -1;
00286     else {
00287         for (i = 1; i <= data.block[0]; i++)
00288             values[i-1] = data.block[i];
00289         return data.block[0];
00290     }
00291 }
00292
00293 static inline __s32 i2c_smbus_write_i2c_block_data(int file, __u8 command,
00294                                                  __u8 length,
00295                                                  const __u8 *values)
00296 {
00297     union i2c_smbus_data data;
00298     int i;
00299     if (length > 32)
00300         length = 32;
00301     for (i = 1; i <= length; i++)
00302         data.block[i] = values[i-1];
00303     data.block[0] = length;
00304     return i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00305                             I2C_SMBUS_I2C_BLOCK_BROKEN, &data);
00306 }
00307
00308 /* Returns the number of read bytes */
00309 static inline __s32 i2c_smbus_block_process_call(int file, __u8 command,
00310                                                  __u8 length, __u8 *values)
00311 {
00312     union i2c_smbus_data data;
00313     int i;
00314     if (length > 32)
00315         length = 32;
00316     for (i = 1; i <= length; i++)
00317         data.block[i] = values[i-1];
00318     data.block[0] = length;
00319     if (i2c_smbus_access(file, I2C_SMBUS_WRITE, command,
00320                          I2C_SMBUS_BLOCK_PROC_CALL, &data))
00321         return -1;
00322     else {
00323         for (i = 1; i <= data.block[0]; i++)
00324             values[i-1] = data.block[i];
00325         return data.block[0];
00326     }
00327 }
00328
00329
00330 #endif /* _LINUX_I2C_DEV_H */

```

7.26 /home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp File Reference

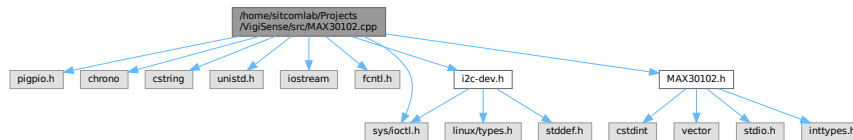
```

#include <pigpio.h>
#include <chrono>

```

```
#include <cstring>
#include <unistd.h>
#include <iostream>
#include <fcntl.h>
#include <sys/ioctl.h>
#include "i2c-dev.h"
#include "MAX30102.h"
```

Include dependency graph for MAX30102.cpp:



Variables

- static const uint8_t REG_INTSTAT1 = 0x00
- static const uint8_t REG_INTSTAT2 = 0x01
- static const uint8_t REG_INTENABLE1 = 0x02
- static const uint8_t REG_INTENABLE2 = 0x03
- static const uint8_t REG_FIFOWRITEPTR = 0x04
- static const uint8_t REG_FIFO_OVERFLOW = 0x05
- static const uint8_t REG_FIFOREADPTR = 0x06
- static const uint8_t REG_FIFO_DATA = 0x07
- static const uint8_t REG_FIFO_CONFIG = 0x08
- static const uint8_t REG_MODE_CONFIG = 0x09
- static const uint8_t REG_PARTICLE_CONFIG = 0x0A
- static const uint8_t REG_LED1_PULSEAMP = 0x0C
- static const uint8_t REG_LED2_PULSEAMP = 0x0D
- static const uint8_t REG_LED_PROX_AMP = 0x10
- static const uint8_t REG_MULTILED_CONFIG1 = 0x11
- static const uint8_t REG_MULTILED_CONFIG2 = 0x12
- static const uint8_t REG_DIETEMP_INT = 0x1F
- static const uint8_t REG_DIETEMP_FRAC = 0x20
- static const uint8_t REG_DIETEMP_CONFIG = 0x21
- static const uint8_t REG_PROX_INT_THRESH = 0x30
- static const uint8_t REG_REVISION_ID = 0xFE
- static const uint8_t REG_PART_ID = 0xFF
- static const uint8_t MAX30102_EXPECTED_PART_ID = 0x15
- static const uint8_t MASK_INT_A_FULL = (uint8_t)~0b10000000
- static const uint8_t INT_A_FULL_ENABLE = 0x80
- static const uint8_t INT_A_FULL_DISABLE = 0x00
- static const uint8_t MASK_INT_DATA_RDY = (uint8_t)~0b01000000
- static const uint8_t INT_DATA_RDY_ENABLE = 0x40
- static const uint8_t INT_DATA_RDY_DISABLE = 0x00
- static const uint8_t MASK_INT_ALC_OVF = (uint8_t)~0b00100000
- static const uint8_t INT_ALC_OVF_ENABLE = 0x20
- static const uint8_t INT_ALC_OVF_DISABLE = 0x00
- static const uint8_t MASK_INT_PROX_INT = (uint8_t)~0b00010000
- static const uint8_t INT_PROX_INT_ENABLE = 0x10
- static const uint8_t INT_PROX_INT_DISABLE = 0x00

- static const uint8_t MASK_INT_DIE_TEMP_RDY = (uint8_t)~0b00000010
- static const uint8_t INT_DIE_TEMP_RDY_ENABLE = 0x02
- static const uint8_t INT_DIE_TEMP_RDY_DISABLE = 0x00
- static const uint8_t MASK_SAMPLEAVG = (uint8_t)~0b11100000
- static const uint8_t SAMPLEAVG_1 = 0x00
- static const uint8_t SAMPLEAVG_2 = 0x20
- static const uint8_t SAMPLEAVG_4 = 0x40
- static const uint8_t SAMPLEAVG_8 = 0x60
- static const uint8_t SAMPLEAVG_16 = 0x80
- static const uint8_t SAMPLEAVG_32 = 0xA0
- static const uint8_t MASK_ROLLOVER = 0xEF
- static const uint8_t ROLLOVER_ENABLE = 0x10
- static const uint8_t ROLLOVER_DISABLE = 0x00
- static const uint8_t MASK_A_FULL = 0xF0
- static const uint8_t MASK_SHUTDOWN = 0x7f
- static const uint8_t SHUTDOWN = 0x80
- static const uint8_t WAKEUP = 0x00
- static const uint8_t MASK_RESET = 0xBF
- static const uint8_t RESET = 0x40
- static const uint8_t MASK_LEDMODE = 0xF8

IR led mode.

- static const uint8_t LEDMODE_REDONLY = 0x02
- static const uint8_t LEDMODE_REDIRONLY = 0x03
- static const uint8_t LEDMODE_MULTILED = 0x07
- static const uint8_t MASK_ADCRANGE = 0x9F
- static const uint8_t ADCRANGE_2048 = 0x00
- static const uint8_t ADCRANGE_4096 = 0x20
- static const uint8_t ADCRANGE_8192 = 0x40
- static const uint8_t ADCRANGE_16384 = 0x60
- static const uint8_t MASK_SAMPLERATE = 0xE3
- static const uint8_t SAMPLERATE_50 = 0x00
- static const uint8_t SAMPLERATE_100 = 0x04
- static const uint8_t SAMPLERATE_200 = 0x08
- static const uint8_t SAMPLERATE_400 = 0x0C
- static const uint8_t SAMPLERATE_800 = 0x10
- static const uint8_t SAMPLERATE_1000 = 0x14
- static const uint8_t SAMPLERATE_1600 = 0x18
- static const uint8_t SAMPLERATE_3200 = 0x1C
- static const uint8_t MASK_PULSEWIDTH = 0xFC
- static const uint8_t PULSEWIDTH_69 = 0x00
- static const uint8_t PULSEWIDTH_118 = 0x01
- static const uint8_t PULSEWIDTH_215 = 0x02
- static const uint8_t PULSEWIDTH_411 = 0x03
- static const uint8_t MASK_SLOT1 = 0xF8
- static const uint8_t MASK_SLOT2 = 0x8F
- static const uint8_t MASK_SLOT3 = 0xF8
- static const uint8_t MASK_SLOT4 = 0x8F
- static const uint8_t SLOT_NONE = 0x00
- static const uint8_t SLOT_RED_LED = 0x01
- static const uint8_t SLOT_IR_LED = 0x02
- static const uint8_t SLOT_NONE_PILOT = 0x04
- static const uint8_t SLOT_RED_PILOT = 0x05
- static const uint8_t SLOT_IR_PILOT = 0x06

7.26.1 Variable Documentation

7.26.1.1 ADCRANGE_16384

```
const uint8_t ADCRANGE_16384 = 0x60 [static]
```

7.26.1.2 ADCRANGE_2048

```
const uint8_t ADCRANGE_2048 = 0x00 [static]
```

7.26.1.3 ADCRANGE_4096

```
const uint8_t ADCRANGE_4096 = 0x20 [static]
```

7.26.1.4 ADCRANGE_8192

```
const uint8_t ADCRANGE_8192 = 0x40 [static]
```

7.26.1.5 INT_A_FULL_DISABLE

```
const uint8_t INT_A_FULL_DISABLE = 0x00 [static]
```

7.26.1.6 INT_A_FULL_ENABLE

```
const uint8_t INT_A_FULL_ENABLE = 0x80 [static]
```

7.26.1.7 INT_ALC_OVF_DISABLE

```
const uint8_t INT_ALC_OVF_DISABLE = 0x00 [static]
```

7.26.1.8 INT_ALC_OVF_ENABLE

```
const uint8_t INT_ALC_OVF_ENABLE = 0x20 [static]
```

7.26.1.9 INT_DATA_RDY_DISABLE

```
const uint8_t INT_DATA_RDY_DISABLE = 0x00 [static]
```

7.26.1.10 INT_DATA_RDY_ENABLE

```
const uint8_t INT_DATA_RDY_ENABLE = 0x40 [static]
```

7.26.1.11 INT_DIE_TEMP_RDY_DISABLE

```
const uint8_t INT_DIE_TEMP_RDY_DISABLE = 0x00 [static]
```

7.26.1.12 INT_DIE_TEMP_RDY_ENABLE

```
const uint8_t INT_DIE_TEMP_RDY_ENABLE = 0x02 [static]
```

7.26.1.13 INT_PROX_INT_DISABLE

```
const uint8_t INT_PROX_INT_DISABLE = 0x00 [static]
```

7.26.1.14 INT_PROX_INT_ENABLE

```
const uint8_t INT_PROX_INT_ENABLE = 0x10 [static]
```

7.26.1.15 LEDMODE_MULTILED

```
const uint8_t LEDMODE_MULTILED = 0x07 [static]
```

7.26.1.16 LEDMODE_REDIRONLY

```
const uint8_t LEDMODE_REDIRONLY = 0x03 [static]
```

7.26.1.17 LEDMODE_REDONLY

```
const uint8_t LEDMODE_REDONLY = 0x02 [static]
```

7.26.1.18 MASK_A_FULL

```
const uint8_t MASK_A_FULL = 0xF0 [static]
```

7.26.1.19 MASK_ADCRANGE

```
const uint8_t MASK_ADCRANGE = 0x9F [static]
```

7.26.1.20 MASK_INT_A_FULL

```
const uint8_t MASK_INT_A_FULL = (uint8_t)~0b10000000 [static]
```

7.26.1.21 MASK_INT_ALC_OVF

```
const uint8_t MASK_INT_ALC_OVF = (uint8_t)~0b00100000 [static]
```

7.26.1.22 MASK_INT_DATA_RDY

```
const uint8_t MASK_INT_DATA_RDY = (uint8_t)~0b01000000 [static]
```

7.26.1.23 MASK_INT_DIE_TEMP_RDY

```
const uint8_t MASK_INT_DIE_TEMP_RDY = (uint8_t)~0b00000010 [static]
```

7.26.1.24 MASK_INT_PROX_INT

```
const uint8_t MASK_INT_PROX_INT = (uint8_t)~0b00010000 [static]
```

7.26.1.25 MASK_LEDMODE

```
const uint8_t MASK_LEDMODE = 0xF8 [static]
```

IR led mode.

7.26.1.26 MASK_PULSEWIDTH

```
const uint8_t MASK_PULSEWIDTH = 0xFC [static]
```

7.26.1.27 MASK_RESET

```
const uint8_t MASK_RESET = 0xBF [static]
```

7.26.1.28 MASK_ROLLOVER

```
const uint8_t MASK_ROLLOVER = 0xEF [static]
```

7.26.1.29 MASK_SAMPLEAVG

```
const uint8_t MASK_SAMPLEAVG = (uint8_t)~0b11100000 [static]
```

7.26.1.30 MASK_SAMPLERATE

```
const uint8_t MASK_SAMPLERATE = 0xE3 [static]
```

7.26.1.31 MASK_SHUTDOWN

```
const uint8_t MASK_SHUTDOWN = 0x7f [static]
```

7.26.1.32 MASK_SLOT1

```
const uint8_t MASK_SLOT1 = 0xF8 [static]
```

7.26.1.33 MASK_SLOT2

```
const uint8_t MASK_SLOT2 = 0x8F [static]
```

7.26.1.34 MASK_SLOT3

```
const uint8_t MASK_SLOT3 = 0xF8 [static]
```

7.26.1.35 MASK_SLOT4

```
const uint8_t MASK_SLOT4 = 0x8F [static]
```

7.26.1.36 MAX30102_EXPECTEDPARTID

```
const uint8_t MAX30102_EXPECTEDPARTID = 0x15 [static]
```

7.26.1.37 PULSEWIDTH_118

```
const uint8_t PULSEWIDTH_118 = 0x01 [static]
```

7.26.1.38 PULSEWIDTH_215

```
const uint8_t PULSEWIDTH_215 = 0x02 [static]
```

7.26.1.39 PULSEWIDTH_411

```
const uint8_t PULSEWIDTH_411 = 0x03 [static]
```

7.26.1.40 PULSEWIDTH_69

```
const uint8_t PULSEWIDTH_69 = 0x00 [static]
```

7.26.1.41 REG_DIETEMPCONFIG

```
const uint8_t REG_DIETEMPCONFIG = 0x21 [static]
```

7.26.1.42 REG_DIETEMPFRAC

```
const uint8_t REG_DIETEMPFRAC = 0x20 [static]
```

7.26.1.43 REG_DIETEMPINT

```
const uint8_t REG_DIETEMPINT = 0x1F [static]
```

7.26.1.44 REG_FIFOCONFIG

```
const uint8_t REG_FIFOCONFIG = 0x08 [static]
```

7.26.1.45 REG_FIFODATA

```
const uint8_t REG_FIFODATA = 0x07 [static]
```

7.26.1.46 REG_FIFOOVERFLOW

```
const uint8_t REG_FIFOOVERFLOW = 0x05 [static]
```

7.26.1.47 REG_FIFOREADPTR

```
const uint8_t REG_FIFOREADPTR = 0x06 [static]
```

7.26.1.48 REG_FIFOWRITEPTR

```
const uint8_t REG_FIFOWRITEPTR = 0x04 [static]
```

7.26.1.49 REG_INTENABLE1

```
const uint8_t REG_INTENABLE1 = 0x02 [static]
```

7.26.1.50 REG_INTENABLE2

```
const uint8_t REG_INTENABLE2 = 0x03 [static]
```

7.26.1.51 REG_INTSTAT1

```
const uint8_t REG_INTSTAT1 = 0x00 [static]
```

7.26.1.52 REG_INTSTAT2

```
const uint8_t REG_INTSTAT2 = 0x01 [static]
```

7.26.1.53 REG_LED1_PULSEAMP

```
const uint8_t REG_LED1_PULSEAMP = 0x0C [static]
```

7.26.1.54 REG_LED2_PULSEAMP

```
const uint8_t REG_LED2_PULSEAMP = 0x0D [static]
```

7.26.1.55 REG_LED_PROX_AMP

```
const uint8_t REG_LED_PROX_AMP = 0x10 [static]
```

7.26.1.56 REG_MODECONFIG

```
const uint8_t REG_MODECONFIG = 0x09 [static]
```

7.26.1.57 REG_MULTILEDCONFIG1

```
const uint8_t REG_MULTILEDCONFIG1 = 0x11 [static]
```

7.26.1.58 REG_MULTILEDCONFIG2

```
const uint8_t REG_MULTILEDCONFIG2 = 0x12 [static]
```

7.26.1.59 REG_PARTICLECONFIG

```
const uint8_t REG_PARTICLECONFIG = 0x0A [static]
```

7.26.1.60 REG_PARTID

```
const uint8_t REG_PARTID = 0xFF [static]
```

7.26.1.61 REG_PROXINTTHRESH

```
const uint8_t REG_PROXINTTHRESH = 0x30 [static]
```

7.26.1.62 REG_REVISIONID

```
const uint8_t REG_REVISIONID = 0xFE [static]
```

7.26.1.63 RESET

```
const uint8_t RESET = 0x40 [static]
```

7.26.1.64 ROLLOVER_DISABLE

```
const uint8_t ROLLOVER_DISABLE = 0x00 [static]
```

7.26.1.65 ROLLOVER_ENABLE

```
const uint8_t ROLLOVER_ENABLE = 0x10 [static]
```

7.26.1.66 SAMPLEAVG_1

```
const uint8_t SAMPLEAVG_1 = 0x00 [static]
```

7.26.1.67 SAMPLEAVG_16

```
const uint8_t SAMPLEAVG_16 = 0x80 [static]
```

7.26.1.68 SAMPLEAVG_2

```
const uint8_t SAMPLEAVG_2 = 0x20 [static]
```

7.26.1.69 SAMPLEAVG_32

```
const uint8_t SAMPLEAVG_32 = 0xA0 [static]
```

7.26.1.70 SAMPLEAVG_4

```
const uint8_t SAMPLEAVG_4 = 0x40 [static]
```


7.26.1.71 SAMPLEAVG_8

```
const uint8_t SAMPLEAVG_8 = 0x60 [static]
```

7.26.1.72 SAMPLERATE_100

```
const uint8_t SAMPLERATE_100 = 0x04 [static]
```

7.26.1.73 SAMPLERATE_1000

```
const uint8_t SAMPLERATE_1000 = 0x14 [static]
```

7.26.1.74 SAMPLERATE_1600

```
const uint8_t SAMPLERATE_1600 = 0x18 [static]
```

7.26.1.75 SAMPLERATE_200

```
const uint8_t SAMPLERATE_200 = 0x08 [static]
```

7.26.1.76 SAMPLERATE_3200

```
const uint8_t SAMPLERATE_3200 = 0x1C [static]
```

7.26.1.77 SAMPLERATE_400

```
const uint8_t SAMPLERATE_400 = 0x0C [static]
```

7.26.1.78 SAMPLERATE_50

```
const uint8_t SAMPLERATE_50 = 0x00 [static]
```

7.26.1.79 SAMPLERATE_800

```
const uint8_t SAMPLERATE_800 = 0x10 [static]
```

7.26.1.80 SHUTDOWN

```
const uint8_t SHUTDOWN = 0x80 [static]
```

7.26.1.81 SLOT_IR_LED

```
const uint8_t SLOT_IR_LED = 0x02 [static]
```

7.26.1.82 SLOT_IR_PILOT

```
const uint8_t SLOT_IR_PILOT = 0x06 [static]
```

7.26.1.83 SLOT_NONE

```
const uint8_t SLOT_NONE = 0x00 [static]
```

7.26.1.84 SLOT_NONE_PILOT

```
const uint8_t SLOT_NONE_PILOT = 0x04 [static]
```

7.26.1.85 SLOT_RED_LED

```
const uint8_t SLOT_RED_LED = 0x01 [static]
```

7.26.1.86 SLOT_RED_PILOT

```
const uint8_t SLOT_RED_PILOT = 0x05 [static]
```

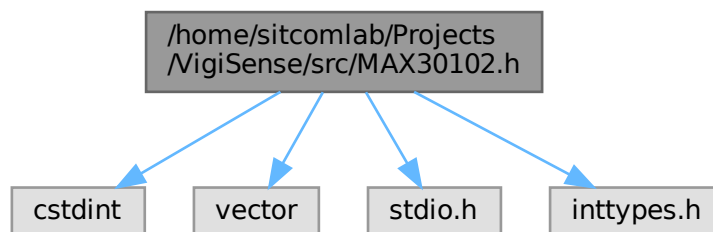
7.26.1.87 WAKEUP

```
const uint8_t WAKEUP = 0x00 [static]
```

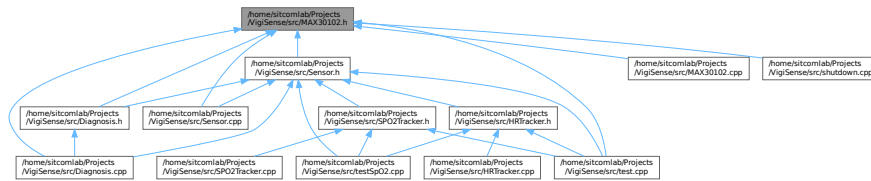
7.27 /home/sitcomlab/Projects/VigiSense/src/MAX30102.h File Reference

```
#include <cstdint>
#include <vector>
#include <stdio.h>
#include <inttypes.h>
```

Include dependency graph for MAX30102.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [MAX30102](#)
- struct [MAX30102::Record](#)

Macros

- `#define` [MAX30102_ADDRESS](#) 0x57
- `#define` [I2C_SPEED_STANDARD](#) 100000
- `#define` [I2C_SPEED_FAST](#) 400000
- `#define` [I2C_BUFFER_LENGTH](#) 32
- `#define` [DEFAULT_INT_GPIO](#) 0
- `#define` [STORAGE_SIZE](#) 4

7.27.1 Macro Definition Documentation

7.27.1.1 DEFAULT_INT_GPIO

```
#define DEFAULT_INT_GPIO 0
```

7.27.1.2 I2C_BUFFER_LENGTH

```
#define I2C_BUFFER_LENGTH 32
```

7.27.1.3 I2C_SPEED_FAST

```
#define I2C_SPEED_FAST 400000
```

7.27.1.4 I2C_SPEED_STANDARD

```
#define I2C_SPEED_STANDARD 100000
```

7.27.1.5 MAX30102_ADDRESS

```
#define MAX30102_ADDRESS 0x57
```

7.27.1.6 STORAGE_SIZE

```
#define STORAGE_SIZE 4
```

7.28 MAX30102.h

[Go to the documentation of this file.](#)

```
00001 /*****
00002  This is a library written for the Maxim MAX30102 Optical Smoke Detector
00003  It should also work with the MAX30102. However, the MAX30102 does not have a Green LED.
00004
00005  These sensors use I2C to communicate, as well as a single (optional)
00006  interrupt line that is not currently supported in this driver.
00007
00008  Written by Garrettluu (https://github.com/garrettluu)
00009  BSD license, all text above must be included in any redistribution.
00010 *****/
00011
00012 /*
00013 Edited by Yu Kit Foo, to include data extraction using Interrupts
00014 Functions changed: changed DEFAULT_INT_GPIO, added hasSample(), dataReady(), gpioISR()
00015 */
00016 #pragma once
00017 #include <cstdint>
00018 #include <vector>
00019 #include <stdio.h>
00020 #include <inttypes.h>
00021
00022 #define MAX30102_ADDRESS    0x57
00023
00024 #define I2C_SPEED_STANDARD  100000
00025 #define I2C_SPEED_FAST      400000
00026
00027 #define I2C_BUFFER_LENGTH   32
00028
00029 // define the GPIO used for the sensor here
00030 #define DEFAULT_INT_GPIO    0
00031
00032 class MAX30102 {
00033     public:
00034         MAX30102(void);
00035         int begin(uint32_t i2cSpeed = I2C_SPEED_STANDARD, uint8_t i2cAddr = MAX30102_ADDRESS);
00036
00037         uint32_t getRed(void); // Returns immediate red value
00038         uint32_t getIR(void); // Returns immediate IR value
00039         bool safeCheck(uint8_t maxTimeToCheck); // Given a max amount of time, checks for new data.
00040
00041         // Configuration
00042         void wakeUp();
00043         void shutDown();
00044         void softReset();
00045
00046         void setLEDMode(uint8_t mode);
00047
00048         void setADCRange(uint8_t adcRange);
00049         void setSampleRate(uint8_t sampleRate);
00050         void setPulseWidth(uint8_t pulseWidth);
00051
00052         void setPulseAmplitudeRed(uint8_t value);
00053         void setPulseAmplitudeIR(uint8_t value);
00054         void setPulseAmplitudeProximity(uint8_t value);
00055
00056         void setProximityThreshold(uint8_t threshMSB);
00057
00058
00059         // Multi-LED configuration mode
00060         void enableSlot(uint8_t slotNumber, uint8_t device);
00061         void disableSlots(void);
00062
00063         // Data Collection
00064
00065         // Interrupts
00066         uint8_t getINT1(void);
00067         uint8_t getINT2(void);
00068         void enableAFULL(void);
00069         void disableAFULL(void);
00070         void enableDATARDY(void);
00071         void disableDATARDY(void);
00072         void enableALCOVF(void);
```

```

00073     void disableALCOVF(void);
00074     void enablePROXINT(void);
00075     void disablePROXINT(void);
00076     void enableDIETEMPRDY(void);
00077     void disableDIETEMPRDY(void);
00078
00079     // FIFO Configurations
00080     void setFIFOAverage(uint8_t samples);
00081     void enableFIFORollover();
00082     void disableFIFORollover();
00083     void setFIFOAlmostFull(uint8_t samples);
00084
00085     // FIFO Reading
00086     uint16_t check(void);
00087     uint8_t available(void);
00088     void nextSample(void);
00089     uint32_t getFIFORed(void);
00090     uint32_t getFIFOIR(void);
00091
00092     uint8_t getWritePointer(void);
00093     uint8_t getReadPointer(void);
00094     void clearFIFO(void);
00095
00096     // Proximity Mode Interrupt Threshold
00097     void setPROXINTTHRESH(uint8_t val);
00098
00099     // Die Temperature
00100     float readTemperature();
00101     float readTemperatureF();
00102
00103     // Detecting ID/Revision
00104     uint8_t getRevisionID();
00105     uint8_t readPartID();
00106
00107
00108     virtual void hasSample();
00109
00110     // Setup the sensor with user selectable settings
00111     void setup(uint8_t powerLevel = 0x1F, uint8_t sampleAverage = 4, uint8_t ledMode = 2, int
sampleRate = 400, int pulseWidth = 411, int adcRange = 4096);
00112     private:
00113         int _i2c;
00114         uint8_t _i2caddr;
00115
00116         uint8_t activeLEDs;
00117
00118         uint8_t revisionID;
00119
00120         void readRevisionID();
00121
00122         void bitMask(uint8_t reg, uint8_t mask, uint8_t thing);
00123
00124         std::vector<uint8_t> readMany(uint8_t address, uint8_t length);
00125
00126         #define STORAGE_SIZE 4
00127         typedef struct Record {
00128             uint32_t red[STORAGE_SIZE];
00129             uint32_t IR[STORAGE_SIZE];
00130             uint8_t head;
00131             uint8_t tail;
00132         } sense_struct;
00133         sense_struct sense;
00134
00135         void dataReady();
00136
00137         static void gpioISR(int, int, uint32_t, void* userdata) {
00138             ((MAX30102*)userdata)->dataReady();
00139         }
00140 };

```

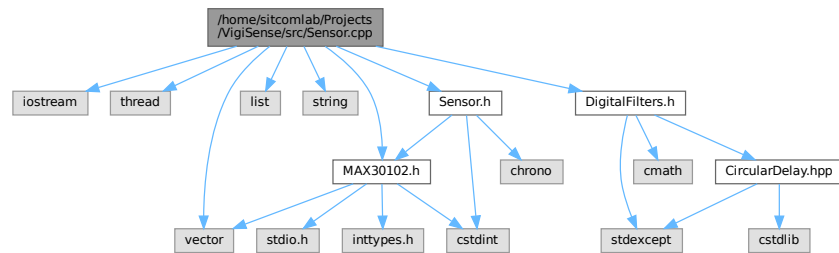
7.29 /home/sitcomlab/Projects/VigiSense/src/Sensor.cpp File Reference

```

#include <iostream>
#include <thread>
#include <vector>
#include <list>
#include <string>
#include "MAX30102.h"

```

```
#include "Sensor.h"
#include "DigitalFilters.h"
Include dependency graph for Sensor.cpp:
```



Variables

- [LowPassFilter lpf](#) (0.08, M_PI)
- [HighPassFilter hpf](#) (0.08, M_PI)
- float [R](#)
- float [SpO2](#)
- bool [crest](#) = false
- bool [trough](#) = false
- uint8_t [dataBeenIncreasing](#) = 0
- uint8_t [nextPastPeaksIndex](#) = 0

7.29.1 Variable Documentation

7.29.1.1 crest

```
bool crest = false
```

7.29.1.2 dataBeenIncreasing

```
uint8_t dataBeenIncreasing = 0
```

7.29.1.3 hpf

```
HighPassFilter hpf(0.08, M_PI) (
    0.08,
    M_PI )
```

7.29.1.4 lpf

```
LowPassFilter lpf(0.08, M_PI) (
    0.08,
    M_PI )
```

7.29.1.5 nextPastPeaksIndex

```
uint8_t nextPastPeaksIndex = 0
```

7.29.1.6 R

```
float R
```

7.29.1.7 SpO2

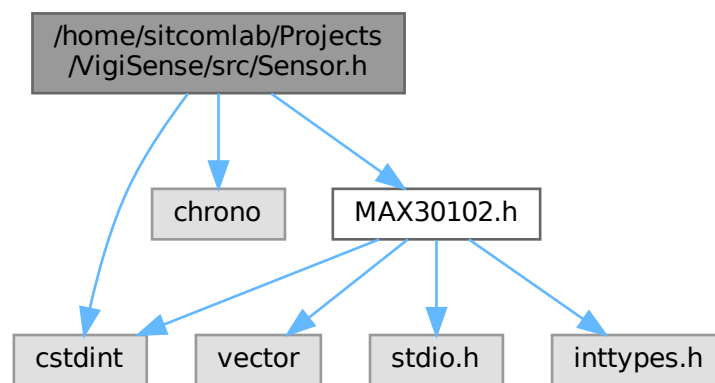
```
float SpO2
```

7.29.1.8 trough

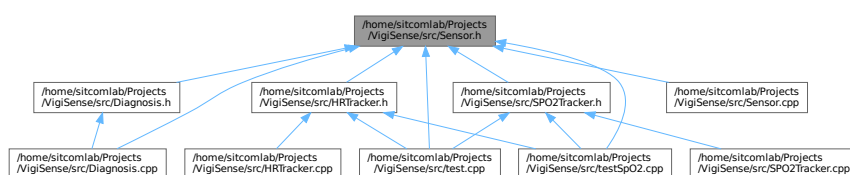
```
bool trough = false
```

7.30 /home/sitcomlab/Projects/VigiSense/src/Sensor.h File Reference

```
#include <cstdint>
#include <chrono>
#include "MAX30102.h"
Include dependency graph for Sensor.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [sensor](#)

7.31 Sensor.h

[Go to the documentation of this file.](#)

```

00001 #pragma once
00002 #include <cstdint>
00003 #include <chrono>
00004 #include "MAX30102.h"
00005
00006 // struct symptomRange{
00007 //     float min;
00008 //     float max;
00009 //     std::string symptom;
00010 // };
00011
00012 //Code refactored from HeartRate.h
00013
00014 class sensor{
00015     public:
00016
00017         sensor(MAX30102 *sensor);
00018         ~sensor();
00019         //Check which functions are fine as is and which need editing/removing
00020         void begin();
00021         void stop();
00022
00023         float getLatestTemperatureF();
00024         void HRcalc();
00025         void stopHRcalc();
00026
00027         // getter for SPO2 and HR
00028         int getSpO2();
00029         int getHR();
00030
00031     protected:
00032         MAX30102* _sensor;
00033         bool running = false;
00034         bool runningHR = false;
00035
00036         //Check which functions are fine as is and which need editing/removing
00037
00038         const int static BPM_BUFFER_SIZE = 4; // Change based on how many samples you want to average
00039         int32_t bpmBuffer[BPM_BUFFER_SIZE];
00040         int nextBPMBufferIndex = 0;
00041
00042         const int static SPO2_BUFFER_SIZE = 4; // Change based on how many samples you want to average
00043         int32_t spo2Buffer[SPO2_BUFFER_SIZE];
00044         int nextSPO2BufferIndex = 0;
00045
00046         std::chrono::time_point<std::chrono::system_clock> timeLastLoopRan;
00047         // IR Data
00048         std::chrono::time_point<std::chrono::system_clock> timeLastIRHeartBeat;
00049         int32_t irLastValue;
00050         int latestIRBPM;
00051         int averageIRBPM;
00052         // Red Data
00053         std::chrono::time_point<std::chrono::system_clock> timeLastRedHeartBeat;
00054         uint64_t redLastValue;
00055         int latestRedBPM;
00056         // Temperature Data
00057         float latestTemperature = -999;
00058
00059         // For Peak Detection
00060         int32_t localMaximaIR;
00061         int32_t localMinimaIR;
00062         int32_t localMaximaRed;
00063         int32_t localMinimaRed;
00064         const static int8_t PAST_PEAKS_SIZE = 2;
00065         int32_t pastMaximasIR[PAST_PEAKS_SIZE];
00066         int32_t pastMinimasIR[PAST_PEAKS_SIZE];
00067         int32_t pastMaximasRed[PAST_PEAKS_SIZE];
00068         int32_t pastMinimasRed[PAST_PEAKS_SIZE];
00069         // SpO2 data
00070         int R;
00071         int latestSpO2;
00072
00073

```



```

00074     void loopThread();
00075     void runHRCalculationLoop();
00076     void updateTemperature();
00077     void resetCalculations();
00078     int32_t Derivative(int32_t data);
00079     int32_t getPeakThreshold();
00080     bool peakDetect(int32_t data);
00081 };
00082
00083 // class spO2Measure : public sensor{
00084 // public:
00085 //     std::vector<symptomRange> symptomRanges {
00086 //         {0,88,"Critically Low Oxygen concentration"},
00087 //         {88,92,"Concerningly Low Oxygen Concentration"},
00088 //         {92,100,"Healthy Oxygen Concentration"}};
00089 //     float critLow = 88;
00090 //     int getSpO2();
00091 //     std::string determineSymptom(float baseline);
00092 // };
00093
00094 // class heartRateMeasure : public sensor{
00095 // public:
00096 //     std::vector<symptomRange> symptomRanges {
00097 //         {0,60,"Bradycardia"},
00098 //         {60,100,"Normal resting heart rate"},
00099 //         {100,200,"Tachycardia"}};
00100 //     float critHigh = 120;
00101 //     float critLow = 40;
00102 //     int getSafeIRHeartRate();
00103 //     int getLatestIRHeartRate();
00104 //     int getLatestRedHeartRate();
00105 //     std::string determineSymptom(float baseline);
00106 // };

```

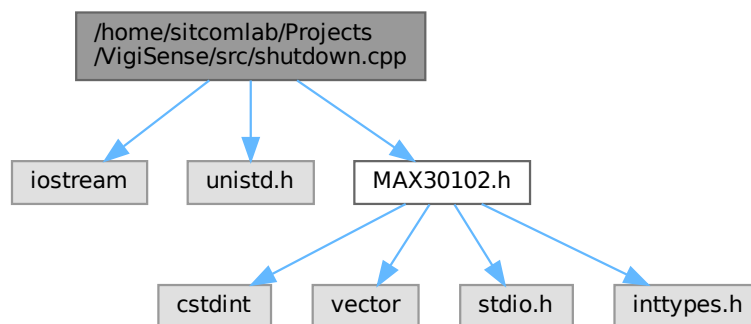
7.32 /home/sitcomlab/Projects/VigiSense/src/shutdown.cpp File Reference

```

#include <iostream>
#include <unistd.h>
#include "MAX30102.h"

```

Include dependency graph for shutdown.cpp:



Functions

- int [main](#) (void)

Variables

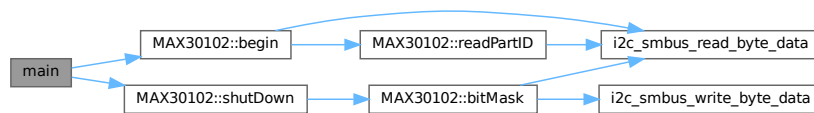
- [MAX30102 heartSensor](#)

7.32.1 Function Documentation

7.32.1.1 main()

```
int main (
    void )
```

Here is the call graph for this function:



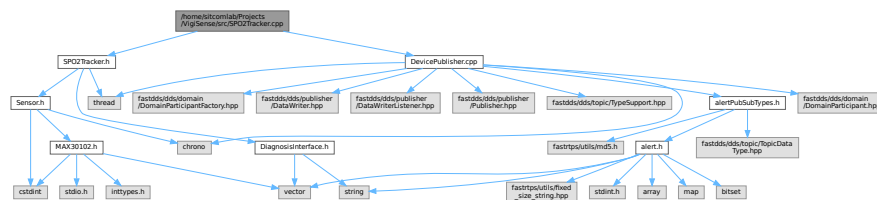
7.32.2 Variable Documentation

7.32.2.1 heartSensor

[MAX30102 heartSensor](#)

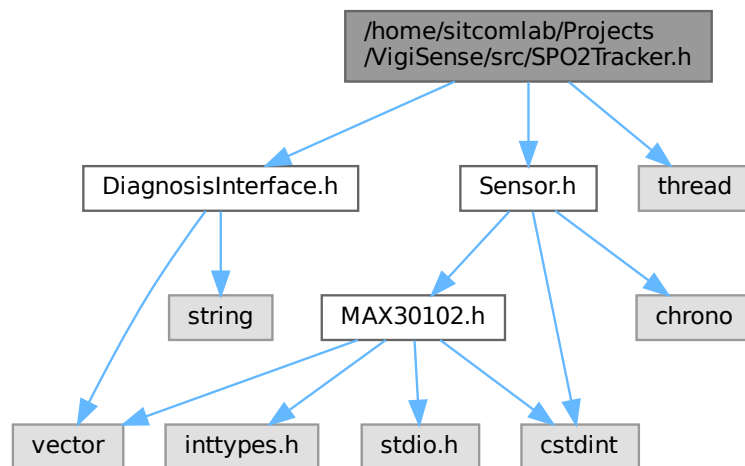
7.33 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp File Reference

```
#include "SPO2Tracker.h"
#include "DevicePublisher.cpp"
Include dependency graph for SPO2Tracker.cpp:
```

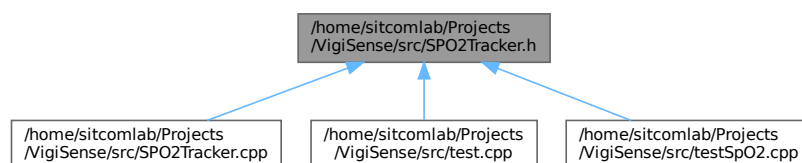


7.34 /home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h File Reference

```
#include "DiagnosisInterface.h"
#include "Sensor.h"
#include <thread>
Include dependency graph for SPO2Tracker.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [SPO2Tracker](#)

7.35 SPO2Tracker.h

[Go to the documentation of this file.](#)

```
00001 #pragma once
00002 #include "DiagnosisInterface.h"
00003 #include "Sensor.h"
```

```

00004 #include <thread>
00005
00006 class SPO2Tracker:public diagnosisInterface {
00007     public:
00008         SPO2Tracker(sensor *s);
00009         ~SPO2Tracker();
00010         void start();
00011         void stop();
00012         void ping();
00013         int getVal();
00014         void tracker();
00015     protected:
00016         sensor* _s;
00017         bool threadRunning = false;
00018         void pingThread();
00019         // define symptom table here
00020         std::vector<symptomRange> symptomRanges {
00021             {0,88,"Critically Low Oxygen concentration"},
00022             {88,92,"Concerningly Low Oxygen Concentration"},
00023             {92,100,"Healthy Oxygen Concentration"}};
00024
00025 };

```

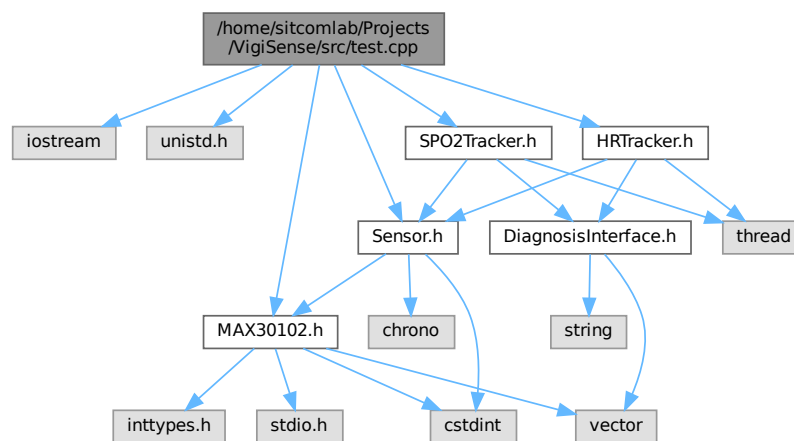
7.36 /home/sitcomlab/Projects/VigiSense/src/test.cpp File Reference

```

#include <iostream>
#include <unistd.h>
#include "MAX30102.h"
#include "Sensor.h"
#include "SPO2Tracker.h"
#include "HRTracker.h"

```

Include dependency graph for test.cpp:



Functions

- int [main](#) (void)

Variables

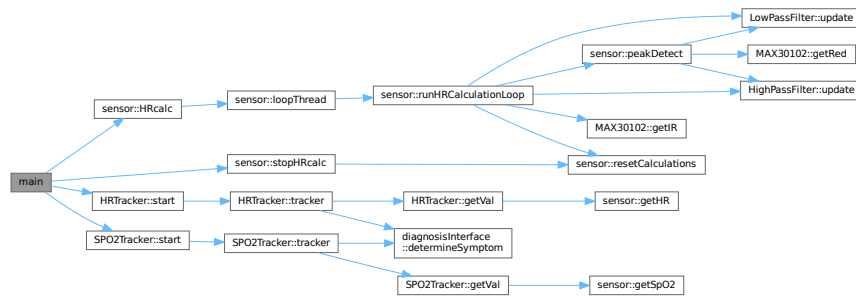
- [MAX30102](#) heartSensor

7.36.1 Function Documentation

7.36.1.1 main()

```
int main (
    void )
```

Here is the call graph for this function:



7.36.2 Variable Documentation

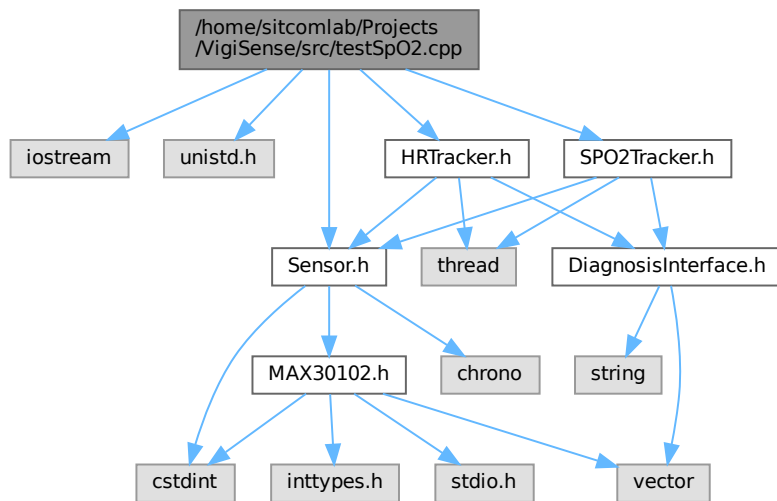
7.36.2.1 heartSensor

MAX30102 heartSensor

7.37 /home/sitcomlab/Projects/VigiSense/src/testSpO2.cpp File Reference

```
#include <iostream>
#include <unistd.h>
#include "Sensor.h"
#include "SPO2Tracker.h"
#include "HRTracker.h"
```

Include dependency graph for testSpO2.cpp:



Functions

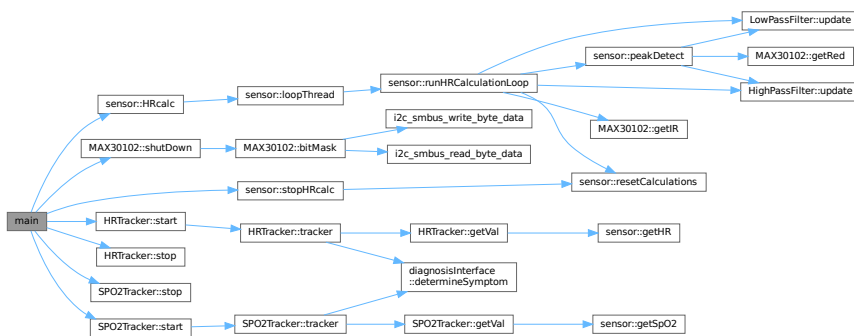
- int [main](#) ()

7.37.1 Function Documentation

7.37.1.1 main()

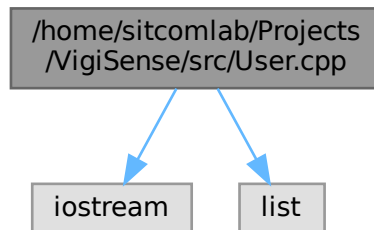
```
int main (
    void )
```

Here is the call graph for this function:



7.38 /home/sitcomlab/Projects/VigiSense/src/User.cpp File Reference

```
#include <iostream>
#include <list>
Include dependency graph for User.cpp:
```



Classes

- struct [contact](#)

Functions

- [User](#) (string *n*)
- [addContact](#) (string *n*, string *e*, long long *p*)

Variables

- struct [contact](#) *name*
- list< [contact](#) > *contacts*

7.38.1 Function Documentation

7.38.1.1 addContact()

```
addContact (
    string n,
    string e,
    long long p )
```

7.38.1.2 User()

```
User (
    string n )
```

7.38.2 Variable Documentation

7.38.2.1 contacts

```
list<contact> contacts
```

7.38.2.2 name

```
struct contact name
```


Index

/home/sitcomlab/Projects/VigiSense/src/CircularDelay.hpp_i2c
128, 129
MAX30102, 97

/home/sitcomlab/Projects/VigiSense/src/DevicePublisher.cpp_i2caddr
131, 132
MAX30102, 97

/home/sitcomlab/Projects/VigiSense/src/DeviceSubscriber.cpp,
134
HRTracker, 56

/home/sitcomlab/Projects/VigiSense/src/Diagnosis.cpp,
135
SPO2Tracker, 117

/home/sitcomlab/Projects/VigiSense/src/Diagnosis.h,
135, 136
_sensor
sensor, 111

/home/sitcomlab/Projects/VigiSense/src/DiagnosisInterface.h,
137, 138
~DevicePublisher
DevicePublisher, 34

/home/sitcomlab/Projects/VigiSense/src/DigitalFilters.h,
138, 141
~DeviceSubscriber
DeviceSubscriber, 37

/home/sitcomlab/Projects/VigiSense/src/HRTracker.cpp,
145
~HRTracker
HRTracker, 54

/home/sitcomlab/Projects/VigiSense/src/HRTracker.h,
145, 146
~PubListener
DevicePublisher::PubListener, 100

/home/sitcomlab/Projects/VigiSense/src/MAX30102.cpp,
161
~SPO2Tracker
SPO2Tracker, 115

/home/sitcomlab/Projects/VigiSense/src/MAX30102.h,
172, 174
~SubListener
DeviceSubscriber::SubListener, 119

/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.cpp,
180
~alert
alert, 12

/home/sitcomlab/Projects/VigiSense/src/SPO2Tracker.h,
181
~alertPubSubType
alertPubSubType, 20

/home/sitcomlab/Projects/VigiSense/src/Sensor.cpp,
175
~sensor
sensor, 107

/home/sitcomlab/Projects/VigiSense/src/Sensor.h, 177,
178
activeLEDs
MAX30102, 97

/home/sitcomlab/Projects/VigiSense/src/User.cpp, 185
ADCRANGE_16384
MAX30102.cpp, 164

/home/sitcomlab/Projects/VigiSense/src/alert.cxx, 121
ADCRANGE_2048
MAX30102.cpp, 164

/home/sitcomlab/Projects/VigiSense/src/alert.h, 121,
123
ADCRANGE_4096
MAX30102.cpp, 164

/home/sitcomlab/Projects/VigiSense/src/alert.idl, 124,
125
ADCRANGE_8192
MAX30102.cpp, 164

/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.cxx,
125
addContact
User.cpp, 185

/home/sitcomlab/Projects/VigiSense/src/alertPubSubTypes.h,
126, 127
addr
i2c_msg, 57

/home/sitcomlab/Projects/VigiSense/src/i2c-dev.h, 147,
158
alert, 11

/home/sitcomlab/Projects/VigiSense/src/shutdown.cpp,
179
~alert, 12

/home/sitcomlab/Projects/VigiSense/src/test.cpp, 182
alert, 12, 13

/home/sitcomlab/Projects/VigiSense/src/testSpO2.cpp,
183
deserialize, 13

_Diagnoses
diagnosis, 39
getCdrSerializedSize, 13

getKeyMaxCdrSerializedSize, 14

getMaxCdrSerializedSize, 14

index, 14, 15, 18

- isKeyDefined, 15
- m_index, 18
- m_message, 18
- message, 15, 16, 18
- operator!=, 16
- operator=, 16
- operator==, 17
- serialize, 17
- serializeKey, 17
- alert.h
 - alert_DllAPI, 123
 - eProsima_user_DllExport, 123
- alert_DllAPI
 - alert.h, 123
- alertPubSubType, 19
 - ~alertPubSubType, 20
 - alertPubSubType, 20
 - createData, 21
 - deleteData, 21
 - deserialize, 21
 - getKey, 21
 - getSerializedSizeProvider, 21
 - m_keyBuffer, 22
 - m_md5, 22
 - serialize, 22
 - type, 20
- alertPubSubTypes.cxx
 - InstanceHandle_t, 125
 - SerializedPayload_t, 125
- amplFac
 - HighPassFilter, 49
 - LowPassFilter3MatchedZ, 77
- available
 - MAX30102, 80
- averageIRBPM
 - sensor, 111
- begin
 - CircularDelay< type, size >, 25
 - MAX30102, 80
 - sensor, 107
- bitMask
 - MAX30102, 80
- block
 - i2c_smbus_data, 58
- BPM_BUFFER_SIZE
 - sensor, 111
- bpmBuffer
 - sensor, 111
- buf
 - i2c_msg, 57
- buffer
 - MovingAvarageFilter< size >, 99
- byte
 - i2c_smbus_data, 58
- calcC_Cr
 - DigitalFilters.h, 140
- check
 - MAX30102, 81
- CircularDelay
 - CircularDelay< type, size >, 24
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 32
 - CircularDelay< type, size >::iterator, 62
 - CircularDelay< type, size >::reverse_iterator, 104
- CircularDelay< type, size >, 23
 - begin, 25
 - CircularDelay, 24
 - data, 26
 - end, 25
 - get, 25
 - push, 25
 - rbegin, 26
 - rend, 26
 - setIterator, 26
- CircularDelay< type, size >::const_iterator, 27
 - CircularDelay, 29
 - const_iterator, 28
 - data_, 29
 - difference_type, 27
 - iterator_category, 27
 - operator!=, 28
 - operator++, 28
 - operator->, 29
 - operator--, 28, 29
 - operator==, 29
 - operator[], 29
 - operator*, 28
 - ptr_, 29
 - self_type, 27
- CircularDelay< type, size >::const_reverse_iterator, 30
 - CircularDelay, 32
 - const_reverse_iterator, 31
 - data_, 33
 - difference_type, 31
 - iterator_category, 31
 - operator!=, 31
 - operator++, 31, 32
 - operator->, 32
 - operator--, 32
 - operator==, 32
 - operator[], 32
 - operator*, 31
 - ptr_, 33
 - self_type, 31
- CircularDelay< type, size >::iterator, 60
 - CircularDelay, 62
 - data_, 62
 - difference_type, 60
 - iterator, 61
 - iterator_category, 60
 - operator!=, 61
 - operator++, 61
 - operator->, 62
 - operator--, 61, 62

- operator==, 62
- operator[], 62
- operator*, 61
- ptr_, 62
- self_type, 60
- CircularDelay< type, size >::reverse_iterator, 102
 - CircularDelay, 104
 - data_, 105
 - difference_type, 103
 - iterator_category, 103
 - operator!=, 103
 - operator++, 103, 104
 - operator->, 104
 - operator--, 104
 - operator==, 104
 - operator[], 104
 - operator*, 103
 - ptr_, 105
 - reverse_iterator, 103
 - self_type, 103
- clearFIFO
 - MAX30102, 81
- command
 - i2c_smbus_ioctl_data, 59
- configOutput
 - HighPassFilter, 48
 - LowPassFilter, 65
 - LowPassFilter2, 68
- const_iterator
 - CircularDelay< type, size >::const_iterator, 28
- const_reverse_iterator
 - CircularDelay< type, size >::const_reverse_iterator, 31
- contact, 33
 - email, 33
 - name, 33
 - phoneNum, 33
- contacts
 - User.cpp, 186
- createData
 - alertPubSubType, 21
- crest
 - Sensor.cpp, 176
- critCheck
 - diagnosis, 39
- CriticalHigh
 - diagnosis, 40
- CriticalLow
 - diagnosis, 40
- critRangeAlert
 - diagnosis, 39
- CustomDiagnoses
 - diagnosis, 40
- CustomDiagnosis
 - diagnosis, 40
- data
 - CircularDelay< type, size >, 26
 - i2c_smbus_ioctl_data, 59
- data_
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 33
 - CircularDelay< type, size >::iterator, 62
 - CircularDelay< type, size >::reverse_iterator, 105
- dataBeenIncreasing
 - Sensor.cpp, 176
- dataReady
 - MAX30102, 81
- DEFAULT_INT_GPIO
 - MAX30102.h, 173
- deleteData
 - alertPubSubType, 21
- Derivative
 - sensor, 107
- deserialize
 - alert, 13
 - alertPubSubType, 21
- determineDiagnosis
 - diagnosis, 39
- determineSymptom
 - diagnosisInterface, 41
- DevicePublisher, 34
 - ~DevicePublisher, 34
 - DevicePublisher, 34
 - init, 35
 - listener_, 35
 - participant_, 35
 - publish, 35
 - publisher_, 35
 - topic_, 35
 - type_, 35
 - writer_, 35
- DevicePublisher::PubListener, 100
 - ~PubListener, 100
 - matched_, 101
 - on_publication_matched, 101
 - PubListener, 100
- DeviceSubscriber, 36
 - ~DeviceSubscriber, 37
 - DeviceSubscriber, 37
 - init, 37
 - listener_, 37
 - participant_, 37
 - reader_, 37
 - subscriber_, 37
 - topic_, 37
 - type_, 37
- DeviceSubscriber::SubListener, 118
 - ~SubListener, 119
 - on_data_available, 119
 - on_subscription_matched, 119
 - SubListener, 119
- Diagnoses
 - diagnosis::DiagnosesTable, 38
- diagnosis, 38
 - _Diagnoses, 39

- critCheck, 39
- CriticalHigh, 40
- CriticalLow, 40
- critRangeAlert, 39
- CustomDiagnoses, 40
- CustomDiagnosis, 40
- determineDiagnosis, 39
- diagnosis::DiagnosisRange, 43
- Diagnosis_Range, 39
- displayDiagnosis, 40
- findMinMax, 40
- SetdiagnosisRanges, 40
- stdDiagnoses, 40
- stdDiagnosis, 40
- diagnosis::DiagnosesTable, 38
 - Diagnoses, 38
- diagnosis::DiagnosisRange, 42
 - diagnosis, 43
 - max, 43
 - min, 43
- Diagnosis_Range
 - diagnosis, 39
- diagnosisInterface, 41
 - determineSymptom, 41
 - getVal, 41
 - ping, 42
 - start, 42
 - stop, 42
 - symptomRanges, 42
- difference_type
 - CircularDelay< type, size >::const_iterator, 27
 - CircularDelay< type, size >::const_reverse_iterator, 31
 - CircularDelay< type, size >::iterator, 60
 - CircularDelay< type, size >::reverse_iterator, 103
- Differentiator
 - Differentiator< T >, 44
- Differentiator< T >, 43
 - Differentiator, 44
 - getOutput, 44
 - sampleTime, 45
 - update, 44
 - x1, 45
 - y, 45
- DigitalFilter< Type >, 45
 - getOutput, 46
 - update, 46
- DigitalFilters.h
 - calcC_Cr, 140
 - squareOf, 140
- disableAFULL
 - MAX30102, 82
- disableALCOVF
 - MAX30102, 82
- disableDATARDY
 - MAX30102, 82
- disableDIETEMPRDY
 - MAX30102, 83
- disableFIFORollover
 - MAX30102, 83
- disablePROXINT
 - MAX30102, 83
- disableSlots
 - MAX30102, 84
- displayDiagnosis
 - diagnosis, 40
- dt
 - HighPassFilter, 49
 - LowPassFilter, 66
- email
 - contact, 33
- enableAFULL
 - MAX30102, 84
- enableALCOVF
 - MAX30102, 84
- enableDATARDY
 - MAX30102, 85
- enableDIETEMPRDY
 - MAX30102, 85
- enableFIFORollover
 - MAX30102, 85
- enablePROXINT
 - MAX30102, 86
- enableSlot
 - MAX30102, 86
- end
 - CircularDelay< type, size >, 25
- epow
 - LowPassFilter, 66
- eprosima, 9
- eprosima::fastcdr, 9
- eProsima_user_DllExport
 - alert.h, 123
- findMinMax
 - diagnosis, 40
- flags
 - i2c_msg, 57
- get
 - CircularDelay< type, size >, 25
- getCdrSerializedSize
 - alert, 13
- getFIFOIR
 - MAX30102, 86
- getFIFOred
 - MAX30102, 87
- getHR
 - sensor, 108
- getINT1
 - MAX30102, 87
- getINT2
 - MAX30102, 87
- getIR
 - MAX30102, 87
- getKey

- alertPubSubType, 21
- getKeyMaxCdrSerializedSize
 - alert, 14
- getLatestTemperatureF
 - sensor, 108
- getMaxCdrSerializedSize
 - alert, 14
- getOutput
 - Differentiator< T >, 44
 - DigitalFilter< Type >, 46
 - HighPassFilter, 48
 - HighPassFilter3, 51
 - LowPassFilter, 65
 - LowPassFilter2, 68
 - LowPassFilter3, 71
 - LowPassFilter3DiffApprox, 74
 - LowPassFilter3MatchedZ, 76
- getPeakThreshold
 - sensor, 108
- getReadPointer
 - MAX30102, 88
- getRed
 - MAX30102, 88
- getRevisionID
 - MAX30102, 88
- getSerializedSizeProvider
 - alertPubSubType, 21
- getSpO2
 - sensor, 108
- getVal
 - diagnosisInterface, 41
 - HRTracker, 54
 - SPO2Tracker, 116
- getWritePointer
 - MAX30102, 88
- gpioISR
 - MAX30102, 88
- hasSample
 - MAX30102, 89
- head
 - MAX30102::Record, 101
- heartSensor
 - shutdown.cpp, 180
 - test.cpp, 183
- HighPassFilter, 46
 - amplFac, 49
 - configOutput, 48
 - dt, 49
 - getOutput, 48
 - HighPassFilter, 47
 - output, 49
 - outputPointer, 48
 - update, 48
 - x1, 49
 - y1c, 49
- HighPassFilter3, 50
 - getOutput, 51
 - HighPassFilter3, 51
- update, 51
 - x, 52
 - xc, 52
 - y, 52
 - yc, 52
- hpf
 - Sensor.cpp, 176
- HRcalc
 - sensor, 108
- HRTracker, 52
 - _s, 56
 - ~HRTracker, 54
 - getVal, 54
 - HRTracker, 54
 - ping, 54
 - pingThread, 55
 - start, 55
 - stop, 55
 - symptomRanges, 56
 - threadRunning, 56
 - tracker, 56
- i2c-dev.h
 - I2C_FUNC_10BIT_ADDR, 149
 - I2C_FUNC_I2C, 149
 - I2C_FUNC_PROTOCOL_MANGLING, 149
 - I2C_FUNC_SMBUS_BLOCK_DATA, 149
 - I2C_FUNC_SMBUS_BLOCK_PROC_CALL, 149
 - I2C_FUNC_SMBUS_BYTE, 149
 - I2C_FUNC_SMBUS_BYTE_DATA, 150
 - I2C_FUNC_SMBUS_HWPEC_CALC, 150
 - I2C_FUNC_SMBUS_I2C_BLOCK, 150
 - I2C_FUNC_SMBUS_PEC, 150
 - I2C_FUNC_SMBUS_PROC_CALL, 150
 - I2C_FUNC_SMBUS_QUICK, 150
 - I2C_FUNC_SMBUS_READ_BLOCK_DATA, 150
 - I2C_FUNC_SMBUS_READ_BYTE, 150
 - I2C_FUNC_SMBUS_READ_BYTE_DATA, 151
 - I2C_FUNC_SMBUS_READ_I2C_BLOCK, 151
 - I2C_FUNC_SMBUS_READ_WORD_DATA, 151
 - I2C_FUNC_SMBUS_WORD_DATA, 151
 - I2C_FUNC_SMBUS_WRITE_BLOCK_DATA, 151
 - I2C_FUNC_SMBUS_WRITE_BYTE, 151
 - I2C_FUNC_SMBUS_WRITE_BYTE_DATA, 151
 - I2C_FUNC_SMBUS_WRITE_I2C_BLOCK, 151
 - I2C_FUNC_SMBUS_WRITE_WORD_DATA, 151
 - I2C_FUNCS, 151
 - I2C_M_IGNORE_NAK, 152
 - I2C_M_NO_RD_ACK, 152
 - I2C_M_NOSTART, 152
 - I2C_M_RD, 152
 - I2C_M_REV_DIR_ADDR, 152
 - I2C_M_TEN, 152
 - I2C_PEC, 152
 - I2C_RDWR_IOCTL_MAX_MSGS, 152
 - I2C_RDWR, 152
 - I2C_RETRIES, 152
 - I2C_SLAVE, 153
 - I2C_SLAVE_FORCE, 153

- I2C_SMBUS, [153](#)
- i2c_smbus_access, [155](#)
- I2C_SMBUS_BLOCK_DATA, [153](#)
- I2C_SMBUS_BLOCK_MAX, [153](#)
- I2C_SMBUS_BLOCK_PROC_CALL, [153](#)
- i2c_smbus_block_process_call, [155](#)
- I2C_SMBUS_BYTE, [153](#)
- I2C_SMBUS_BYTE_DATA, [153](#)
- I2C_SMBUS_I2C_BLOCK_BROKEN, [153](#)
- I2C_SMBUS_I2C_BLOCK_DATA, [154](#)
- I2C_SMBUS_I2C_BLOCK_MAX, [154](#)
- I2C_SMBUS_PROC_CALL, [154](#)
- i2c_smbus_process_call, [155](#)
- I2C_SMBUS_QUICK, [154](#)
- I2C_SMBUS_READ, [154](#)
- i2c_smbus_read_block_data, [155](#)
- i2c_smbus_read_byte, [155](#)
- i2c_smbus_read_byte_data, [156](#)
- i2c_smbus_read_i2c_block_data, [156](#)
- i2c_smbus_read_word_data, [156](#)
- I2C_SMBUS_WORD_DATA, [154](#)
- I2C_SMBUS_WRITE, [154](#)
- i2c_smbus_write_block_data, [156](#)
- i2c_smbus_write_byte, [156](#)
- i2c_smbus_write_byte_data, [157](#)
- i2c_smbus_write_i2c_block_data, [157](#)
- i2c_smbus_write_quick, [157](#)
- i2c_smbus_write_word_data, [157](#)
- I2C_TENBIT, [154](#)
- I2C_TIMEOUT, [154](#)
- I2C_BUFFER_LENGTH
 - MAX30102.h, [173](#)
- I2C_FUNC_10BIT_ADDR
 - i2c-dev.h, [149](#)
- I2C_FUNC_I2C
 - i2c-dev.h, [149](#)
- I2C_FUNC_PROTOCOL_MANGLING
 - i2c-dev.h, [149](#)
- I2C_FUNC_SMBUS_BLOCK_DATA
 - i2c-dev.h, [149](#)
- I2C_FUNC_SMBUS_BLOCK_PROC_CALL
 - i2c-dev.h, [149](#)
- I2C_FUNC_SMBUS_BYTE
 - i2c-dev.h, [149](#)
- I2C_FUNC_SMBUS_BYTE_DATA
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_HWPEC_CALC
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_I2C_BLOCK
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_PEC
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_PROC_CALL
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_QUICK
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_READ_BLOCK_DATA
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_READ_BYTE
 - i2c-dev.h, [150](#)
- I2C_FUNC_SMBUS_READ_BYTE_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_READ_I2C_BLOCK
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_READ_WORD_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WORD_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WRITE_BLOCK_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WRITE_BYTE
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WRITE_BYTE_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WRITE_I2C_BLOCK
 - i2c-dev.h, [151](#)
- I2C_FUNC_SMBUS_WRITE_WORD_DATA
 - i2c-dev.h, [151](#)
- I2C_FUNCS
 - i2c-dev.h, [151](#)
- I2C_M_IGNORE_NAK
 - i2c-dev.h, [152](#)
- I2C_M_NO_RD_ACK
 - i2c-dev.h, [152](#)
- I2C_M_NOSTART
 - i2c-dev.h, [152](#)
- I2C_M_RD
 - i2c-dev.h, [152](#)
- I2C_M_REV_DIR_ADDR
 - i2c-dev.h, [152](#)
- I2C_M_TEN
 - i2c-dev.h, [152](#)
- i2c_msg, [57](#)
 - addr, [57](#)
 - buf, [57](#)
 - flags, [57](#)
 - len, [57](#)
- I2C_PEC
 - i2c-dev.h, [152](#)
- I2C_RDRW_IOCTL_MAX_MSGS
 - i2c-dev.h, [152](#)
- I2C_RDWR
 - i2c-dev.h, [152](#)
- i2c_rdwr_ioctl_data, [57](#)
 - msgs, [58](#)
 - nmsgs, [58](#)
- I2C_RETRIES
 - i2c-dev.h, [152](#)
- I2C_SLAVE
 - i2c-dev.h, [153](#)
- I2C_SLAVE_FORCE
 - i2c-dev.h, [153](#)
- I2C_SMBUS
 - i2c-dev.h, [153](#)
- i2c_smbus_access
 - i2c-dev.h, [155](#)

- I2C_SMBUS_BLOCK_DATA
 - i2c-dev.h, [153](#)
- I2C_SMBUS_BLOCK_MAX
 - i2c-dev.h, [153](#)
- I2C_SMBUS_BLOCK_PROC_CALL
 - i2c-dev.h, [153](#)
- i2c_smbus_block_process_call
 - i2c-dev.h, [155](#)
- I2C_SMBUS_BYTE
 - i2c-dev.h, [153](#)
- I2C_SMBUS_BYTE_DATA
 - i2c-dev.h, [153](#)
- i2c_smbus_data, [58](#)
 - block, [58](#)
 - byte, [58](#)
 - word, [58](#)
- I2C_SMBUS_I2C_BLOCK_BROKEN
 - i2c-dev.h, [153](#)
- I2C_SMBUS_I2C_BLOCK_DATA
 - i2c-dev.h, [154](#)
- I2C_SMBUS_I2C_BLOCK_MAX
 - i2c-dev.h, [154](#)
- i2c_smbus_ioctl_data, [59](#)
 - command, [59](#)
 - data, [59](#)
 - read_write, [59](#)
 - size, [59](#)
- I2C_SMBUS_PROC_CALL
 - i2c-dev.h, [154](#)
- i2c_smbus_process_call
 - i2c-dev.h, [155](#)
- I2C_SMBUS_QUICK
 - i2c-dev.h, [154](#)
- I2C_SMBUS_READ
 - i2c-dev.h, [154](#)
- i2c_smbus_read_block_data
 - i2c-dev.h, [155](#)
- i2c_smbus_read_byte
 - i2c-dev.h, [155](#)
- i2c_smbus_read_byte_data
 - i2c-dev.h, [156](#)
- i2c_smbus_read_i2c_block_data
 - i2c-dev.h, [156](#)
- i2c_smbus_read_word_data
 - i2c-dev.h, [156](#)
- I2C_SMBUS_WORD_DATA
 - i2c-dev.h, [154](#)
- I2C_SMBUS_WRITE
 - i2c-dev.h, [154](#)
- i2c_smbus_write_block_data
 - i2c-dev.h, [156](#)
- i2c_smbus_write_byte
 - i2c-dev.h, [156](#)
- i2c_smbus_write_byte_data
 - i2c-dev.h, [157](#)
- i2c_smbus_write_i2c_block_data
 - i2c-dev.h, [157](#)
- i2c_smbus_write_quick
 - i2c-dev.h, [157](#)
- i2c_smbus_write_word_data
 - i2c-dev.h, [157](#)
- I2C_SPEED_FAST
 - MAX30102.h, [173](#)
- I2C_SPEED_STANDARD
 - MAX30102.h, [173](#)
- I2C_TENBIT
 - i2c-dev.h, [154](#)
- I2C_TIMEOUT
 - i2c-dev.h, [154](#)
- index
 - alert, [14](#), [15](#), [18](#)
- init
 - DevicePublisher, [35](#)
 - DeviceSubscriber, [37](#)
- InstanceHandle_t
 - alertPubSubTypes.cxx, [125](#)
- INT_A_FULL_DISABLE
 - MAX30102.cpp, [164](#)
- INT_A_FULL_ENABLE
 - MAX30102.cpp, [164](#)
- INT_ALC_OVF_DISABLE
 - MAX30102.cpp, [164](#)
- INT_ALC_OVF_ENABLE
 - MAX30102.cpp, [164](#)
- INT_DATA_RDY_DISABLE
 - MAX30102.cpp, [164](#)
- INT_DATA_RDY_ENABLE
 - MAX30102.cpp, [164](#)
- INT_DIE_TEMP_RDY_DISABLE
 - MAX30102.cpp, [164](#)
- INT_DIE_TEMP_RDY_ENABLE
 - MAX30102.cpp, [165](#)
- INT_PROX_INT_DISABLE
 - MAX30102.cpp, [165](#)
- INT_PROX_INT_ENABLE
 - MAX30102.cpp, [165](#)
- IR
 - MAX30102::Record, [101](#)
- irLastValue
 - sensor, [111](#)
- isKeyDefined
 - alert, [15](#)
- iterator
 - CircularDelay< type, size >::iterator, [61](#)
- iterator_category
 - CircularDelay< type, size >::const_iterator, [27](#)
 - CircularDelay< type, size >::const_reverse_iterator, [31](#)
 - CircularDelay< type, size >::iterator, [60](#)
 - CircularDelay< type, size >::reverse_iterator, [103](#)
- latestIRBPM
 - sensor, [111](#)
- latestRedBPM
 - sensor, [111](#)
- latestSpO2
 - sensor, [111](#)

- latestTemperature
 - sensor, [111](#)
- LEDMODE_MULTILED
 - MAX30102.cpp, [165](#)
- LEDMODE_REDIRONL
 - MAX30102.cpp, [165](#)
- LEDMODE_REDONLY
 - MAX30102.cpp, [165](#)
- len
 - i2c_msg, [57](#)
- listener_
 - DevicePublisher, [35](#)
 - DeviceSubscriber, [37](#)
- localMaximaIR
 - sensor, [111](#)
- localMaximaRed
 - sensor, [111](#)
- localMinimalR
 - sensor, [112](#)
- localMinimaRed
 - sensor, [112](#)
- loopThread
 - sensor, [108](#)
- LowPassFilter, [63](#)
 - configOutput, [65](#)
 - dt, [66](#)
 - epow, [66](#)
 - getOutput, [65](#)
 - LowPassFilter, [64](#)
 - output, [66](#)
 - outputPointer, [65](#)
 - update, [65](#)
- LowPassFilter2, [66](#)
 - configOutput, [68](#)
 - getOutput, [68](#)
 - LowPassFilter2, [67](#)
 - update, [68](#)
 - x, [69](#)
 - xc, [69](#)
 - y, [69](#)
 - yc, [69](#)
- LowPassFilter3, [70](#)
 - getOutput, [71](#)
 - LowPassFilter3, [71](#)
 - update, [71](#)
 - x, [72](#)
 - xc, [72](#)
 - y, [72](#)
 - yc, [72](#)
- LowPassFilter3DiffApprox, [72](#)
 - getOutput, [74](#)
 - LowPassFilter3DiffApprox, [73](#)
 - update, [74](#)
 - x, [74](#)
 - xc, [74](#)
 - y, [74](#)
 - yc, [75](#)
- LowPassFilter3MatchedZ, [75](#)
 - amplFac, [77](#)
 - getOutput, [76](#)
 - LowPassFilter3MatchedZ, [76](#)
 - update, [76](#)
 - y, [77](#)
 - yc, [77](#)
- lpf
 - Sensor.cpp, [176](#)
- m_index
 - alert, [18](#)
- m_keyBuffer
 - alertPubSubType, [22](#)
- m_md5
 - alertPubSubType, [22](#)
- m_message
 - alert, [18](#)
- main
 - shutdown.cpp, [180](#)
 - test.cpp, [183](#)
 - testSpO2.cpp, [184](#)
- MASK_A_FULL
 - MAX30102.cpp, [165](#)
- MASK_ADCRANGE
 - MAX30102.cpp, [165](#)
- MASK_INT_A_FULL
 - MAX30102.cpp, [165](#)
- MASK_INT_ALC_OVF
 - MAX30102.cpp, [165](#)
- MASK_INT_DATA_RDY
 - MAX30102.cpp, [166](#)
- MASK_INT_DIE_TEMP_RDY
 - MAX30102.cpp, [166](#)
- MASK_INT_PROX_INT
 - MAX30102.cpp, [166](#)
- MASK_LEDMODE
 - MAX30102.cpp, [166](#)
- MASK_PULSEWIDTH
 - MAX30102.cpp, [166](#)
- MASK_RESET
 - MAX30102.cpp, [166](#)
- MASK_ROLLOVER
 - MAX30102.cpp, [166](#)
- MASK_SAMPLEAVG
 - MAX30102.cpp, [166](#)
- MASK_SAMPLERATE
 - MAX30102.cpp, [166](#)
- MASK_SHUTDOWN
 - MAX30102.cpp, [166](#)
- MASK_SLOT1
 - MAX30102.cpp, [167](#)
- MASK_SLOT2
 - MAX30102.cpp, [167](#)
- MASK_SLOT3
 - MAX30102.cpp, [167](#)
- MASK_SLOT4
 - MAX30102.cpp, [167](#)
- matched_
 - DevicePublisher::PubListener, [101](#)

- max
 - diagnosis::DiagnosisRange, [43](#)
 - symptomRange, [120](#)
- MAX30102, [78](#)
 - _i2c, [97](#)
 - _i2caddr, [97](#)
 - activeLEDs, [97](#)
 - available, [80](#)
 - begin, [80](#)
 - bitMask, [80](#)
 - check, [81](#)
 - clearFIFO, [81](#)
 - dataReady, [81](#)
 - disableAFULL, [82](#)
 - disableALCOVF, [82](#)
 - disableDATARDY, [82](#)
 - disableDIETEMPRDY, [83](#)
 - disableFIFORollover, [83](#)
 - disablePROXINT, [83](#)
 - disableSlots, [84](#)
 - enableAFULL, [84](#)
 - enableALCOVF, [84](#)
 - enableDATARDY, [85](#)
 - enableDIETEMPRDY, [85](#)
 - enableFIFORollover, [85](#)
 - enablePROXINT, [86](#)
 - enableSlot, [86](#)
 - getFIFOIR, [86](#)
 - getFIFORed, [87](#)
 - getINT1, [87](#)
 - getINT2, [87](#)
 - getIR, [87](#)
 - getReadPointer, [88](#)
 - getRed, [88](#)
 - getRevisionID, [88](#)
 - getWritePointer, [88](#)
 - gpioISR, [88](#)
 - hasSample, [89](#)
 - MAX30102, [80](#)
 - nextSample, [89](#)
 - readMany, [89](#)
 - readPartID, [90](#)
 - readRevisionID, [90](#)
 - readTemperature, [90](#)
 - readTemperatureF, [91](#)
 - revisionID, [97](#)
 - safeCheck, [91](#)
 - sense, [97](#)
 - sense_struct, [80](#)
 - setADCRange, [91](#)
 - setFIFOAlmostFull, [92](#)
 - setFIFOAverage, [92](#)
 - setLEDMode, [92](#)
 - setProximityThreshold, [93](#)
 - setPROXINTTHRESH, [93](#)
 - setPulseAmplitudeIR, [93](#)
 - setPulseAmplitudeProximity, [94](#)
 - setPulseAmplitudeRed, [94](#)
 - setPulseWidth, [94](#)
 - setSampleRate, [95](#)
 - setup, [95](#)
 - shutDown, [96](#)
 - softReset, [96](#)
 - wakeUp, [97](#)
- MAX30102.cpp
 - ADCRANGE_16384, [164](#)
 - ADCRANGE_2048, [164](#)
 - ADCRANGE_4096, [164](#)
 - ADCRANGE_8192, [164](#)
 - INT_A_FULL_DISABLE, [164](#)
 - INT_A_FULL_ENABLE, [164](#)
 - INT_ALC_OVF_DISABLE, [164](#)
 - INT_ALC_OVF_ENABLE, [164](#)
 - INT_DATA_RDY_DISABLE, [164](#)
 - INT_DATA_RDY_ENABLE, [164](#)
 - INT_DIE_TEMP_RDY_DISABLE, [164](#)
 - INT_DIE_TEMP_RDY_ENABLE, [165](#)
 - INT_PROX_INT_DISABLE, [165](#)
 - INT_PROX_INT_ENABLE, [165](#)
 - LEDMODE_MULTILED, [165](#)
 - LEDMODE_REDIRONLY, [165](#)
 - LEDMODE_REDONLY, [165](#)
 - MASK_A_FULL, [165](#)
 - MASK_ADCRANGE, [165](#)
 - MASK_INT_A_FULL, [165](#)
 - MASK_INT_ALC_OVF, [165](#)
 - MASK_INT_DATA_RDY, [166](#)
 - MASK_INT_DIE_TEMP_RDY, [166](#)
 - MASK_INT_PROX_INT, [166](#)
 - MASK_LEDMODE, [166](#)
 - MASK_PULSEWIDTH, [166](#)
 - MASK_RESET, [166](#)
 - MASK_ROLLOVER, [166](#)
 - MASK_SAMPLEAVG, [166](#)
 - MASK_SAMPLERATE, [166](#)
 - MASK_SHUTDOWN, [166](#)
 - MASK_SLOT1, [167](#)
 - MASK_SLOT2, [167](#)
 - MASK_SLOT3, [167](#)
 - MASK_SLOT4, [167](#)
 - MAX30102_EXPECTEDPARTID, [167](#)
 - PULSEWIDTH_118, [167](#)
 - PULSEWIDTH_215, [167](#)
 - PULSEWIDTH_411, [167](#)
 - PULSEWIDTH_69, [167](#)
 - REG_DIETEMPCONFIG, [167](#)
 - REG_DIETEMPFRAC, [168](#)
 - REG_DIETEMPINT, [168](#)
 - REG_FIFOCONFIG, [168](#)
 - REG_FIFODATA, [168](#)
 - REG_FIFOOVERFLOW, [168](#)
 - REG_FIFOREADPTR, [168](#)
 - REG_FIFOWRITEPTR, [168](#)
 - REG_INTENABLE1, [168](#)
 - REG_INTENABLE2, [168](#)
 - REG_INTSTAT1, [168](#)

- REG_INTSTAT2, 169
- REG_LED1_PULSEAMP, 169
- REG_LED2_PULSEAMP, 169
- REG_LED_PROX_AMP, 169
- REG_MODECONFIG, 169
- REG_MULTILEDCONFIG1, 169
- REG_MULTILEDCONFIG2, 169
- REG_PARTICLECONFIG, 169
- REG_PARTID, 169
- REG_PROXINTTHRESH, 169
- REG_REVISIONID, 170
- RESET, 170
- ROLLOVER_DISABLE, 170
- ROLLOVER_ENABLE, 170
- SAMPLEAVG_1, 170
- SAMPLEAVG_16, 170
- SAMPLEAVG_2, 170
- SAMPLEAVG_32, 170
- SAMPLEAVG_4, 170
- SAMPLEAVG_8, 170
- SAMPLERATE_100, 171
- SAMPLERATE_1000, 171
- SAMPLERATE_1600, 171
- SAMPLERATE_200, 171
- SAMPLERATE_3200, 171
- SAMPLERATE_400, 171
- SAMPLERATE_50, 171
- SAMPLERATE_800, 171
- SHUTDOWN, 171
- SLOT_IR_LED, 171
- SLOT_IR_PILOT, 172
- SLOT_NONE, 172
- SLOT_NONE_PILOT, 172
- SLOT_RED_LED, 172
- SLOT_RED_PILOT, 172
- WAKEUP, 172
- MAX30102.h
 - DEFAULT_INT_GPIO, 173
 - I2C_BUFFER_LENGTH, 173
 - I2C_SPEED_FAST, 173
 - I2C_SPEED_STANDARD, 173
 - MAX30102_ADDRESS, 173
 - STORAGE_SIZE, 173
- MAX30102::Record, 101
 - head, 101
 - IR, 101
 - red, 101
 - tail, 101
- MAX30102_ADDRESS
 - MAX30102.h, 173
- MAX30102_EXPECTEDPARTID
 - MAX30102.cpp, 167
- message
 - alert, 15, 16, 18
- min
 - diagnosis::DiagnosisRange, 43
 - symptomRange, 120
- MovingAvarageFilter< size >, 98
 - buffer, 99
 - output, 99
 - update, 99
- msgs
 - i2c_rdwr_ioctl_data, 58
- name
 - contact, 33
 - User.cpp, 186
- nextBPMBufferIndex
 - sensor, 112
- nextPastPeaksIndex
 - Sensor.cpp, 176
- nextSample
 - MAX30102, 89
- nextSPO2BufferIndex
 - sensor, 112
- nmsgs
 - i2c_rdwr_ioctl_data, 58
- on_data_available
 - DeviceSubscriber::SubListener, 119
- on_publication_matched
 - DevicePublisher::PubListener, 101
- on_subscription_matched
 - DeviceSubscriber::SubListener, 119
- operator!=
 - alert, 16
 - CircularDelay< type, size >::const_iterator, 28
 - CircularDelay< type, size >::const_reverse_iterator, 31
 - CircularDelay< type, size >::iterator, 61
 - CircularDelay< type, size >::reverse_iterator, 103
- operator++
 - CircularDelay< type, size >::const_iterator, 28
 - CircularDelay< type, size >::const_reverse_iterator, 31, 32
 - CircularDelay< type, size >::iterator, 61
 - CircularDelay< type, size >::reverse_iterator, 103, 104
- operator->
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 32
 - CircularDelay< type, size >::iterator, 62
 - CircularDelay< type, size >::reverse_iterator, 104
- operator--
 - CircularDelay< type, size >::const_iterator, 28, 29
 - CircularDelay< type, size >::const_reverse_iterator, 32
 - CircularDelay< type, size >::iterator, 61, 62
 - CircularDelay< type, size >::reverse_iterator, 104
- operator=
 - alert, 16
- operator==
 - alert, 17
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 32

- CircularDelay< type, size >::iterator, 62
- CircularDelay< type, size >::reverse_iterator, 104
- operator[]
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 32
 - CircularDelay< type, size >::iterator, 62
 - CircularDelay< type, size >::reverse_iterator, 104
- operator*
 - CircularDelay< type, size >::const_iterator, 28
 - CircularDelay< type, size >::const_reverse_iterator, 31
 - CircularDelay< type, size >::iterator, 61
 - CircularDelay< type, size >::reverse_iterator, 103
- output
 - HighPassFilter, 49
 - LowPassFilter, 66
 - MovingAvarageFilter< size >, 99
- outputPointer
 - HighPassFilter, 48
 - LowPassFilter, 65
- participant_
 - DevicePublisher, 35
 - DeviceSubscriber, 37
- PAST_PEAKS_SIZE
 - sensor, 112
- pastMaximasIR
 - sensor, 112
- pastMaximasRed
 - sensor, 112
- pastMinimasIR
 - sensor, 112
- pastMinimasRed
 - sensor, 112
- peakDetect
 - sensor, 109
- phoneNum
 - contact, 33
- ping
 - diagnosisInterface, 42
 - HRTracker, 54
 - SPO2Tracker, 116
- pingThread
 - HRTracker, 55
 - SPO2Tracker, 116
- pow
 - tps, 9
- ptr_
 - CircularDelay< type, size >::const_iterator, 29
 - CircularDelay< type, size >::const_reverse_iterator, 33
 - CircularDelay< type, size >::iterator, 62
 - CircularDelay< type, size >::reverse_iterator, 105
- publish
 - DevicePublisher, 35
- publisher_
 - DevicePublisher, 35
- PubListener
 - DevicePublisher::PubListener, 100
- PULSEWIDTH_118
 - MAX30102.cpp, 167
- PULSEWIDTH_215
 - MAX30102.cpp, 167
- PULSEWIDTH_411
 - MAX30102.cpp, 167
- PULSEWIDTH_69
 - MAX30102.cpp, 167
- push
 - CircularDelay< type, size >, 25
- R
 - sensor, 112
 - Sensor.cpp, 177
- rbegin
 - CircularDelay< type, size >, 26
- read_write
 - i2c_smbus_ioctl_data, 59
- reader_
 - DeviceSubscriber, 37
- readMany
 - MAX30102, 89
- readPartID
 - MAX30102, 90
- readRevisionID
 - MAX30102, 90
- readTemperature
 - MAX30102, 90
- readTemperatureF
 - MAX30102, 91
- red
 - MAX30102::Record, 101
- redLastValue
 - sensor, 113
- REG_DIETEMPCONFIG
 - MAX30102.cpp, 167
- REG_DIETEMPFRAC
 - MAX30102.cpp, 168
- REG_DIETEMPINT
 - MAX30102.cpp, 168
- REG_FIFOCONFIG
 - MAX30102.cpp, 168
- REG_FIFODATA
 - MAX30102.cpp, 168
- REG_FIFOOVERFLOW
 - MAX30102.cpp, 168
- REG_FIFOREADPTR
 - MAX30102.cpp, 168
- REG_FIFOWRITEPTR
 - MAX30102.cpp, 168
- REG_INTENABLE1
 - MAX30102.cpp, 168
- REG_INTENABLE2
 - MAX30102.cpp, 168
- REG_INTSTAT1
 - MAX30102.cpp, 168
- REG_INTSTAT2
 - MAX30102.cpp, 169

- REG_LED1_PULSEAMP
 - MAX30102.cpp, [169](#)
- REG_LED2_PULSEAMP
 - MAX30102.cpp, [169](#)
- REG_LED_PROX_AMP
 - MAX30102.cpp, [169](#)
- REG_MODECONFIG
 - MAX30102.cpp, [169](#)
- REG_MULTILEDCONFIG1
 - MAX30102.cpp, [169](#)
- REG_MULTILEDCONFIG2
 - MAX30102.cpp, [169](#)
- REG_PARTICLECONFIG
 - MAX30102.cpp, [169](#)
- REG_PARTID
 - MAX30102.cpp, [169](#)
- REG_PROXINTTHRESH
 - MAX30102.cpp, [169](#)
- REG_REVISIONID
 - MAX30102.cpp, [170](#)
- rend
 - CircularDelay< type, size >, [26](#)
- RESET
 - MAX30102.cpp, [170](#)
- resetCalculations
 - sensor, [109](#)
- reverse_iterator
 - CircularDelay< type, size >::reverse_iterator, [103](#)
- revisionID
 - MAX30102, [97](#)
- ROLLOVER_DISABLE
 - MAX30102.cpp, [170](#)
- ROLLOVER_ENABLE
 - MAX30102.cpp, [170](#)
- runHRCalculationLoop
 - sensor, [109](#)
- running
 - sensor, [113](#)
- runningHR
 - sensor, [113](#)
- safeCheck
 - MAX30102, [91](#)
- SAMPLEAVG_1
 - MAX30102.cpp, [170](#)
- SAMPLEAVG_16
 - MAX30102.cpp, [170](#)
- SAMPLEAVG_2
 - MAX30102.cpp, [170](#)
- SAMPLEAVG_32
 - MAX30102.cpp, [170](#)
- SAMPLEAVG_4
 - MAX30102.cpp, [170](#)
- SAMPLEAVG_8
 - MAX30102.cpp, [170](#)
- SAMPLERATE_100
 - MAX30102.cpp, [171](#)
- SAMPLERATE_1000
 - MAX30102.cpp, [171](#)
- SAMPLERATE_1600
 - MAX30102.cpp, [171](#)
- SAMPLERATE_200
 - MAX30102.cpp, [171](#)
- SAMPLERATE_3200
 - MAX30102.cpp, [171](#)
- SAMPLERATE_400
 - MAX30102.cpp, [171](#)
- SAMPLERATE_50
 - MAX30102.cpp, [171](#)
- SAMPLERATE_800
 - MAX30102.cpp, [171](#)
- sampleTime
 - Differentiator< T >, [45](#)
- self_type
 - CircularDelay< type, size >::const_iterator, [27](#)
 - CircularDelay< type, size >::const_reverse_iterator, [31](#)
 - CircularDelay< type, size >::iterator, [60](#)
 - CircularDelay< type, size >::reverse_iterator, [103](#)
- sense
 - MAX30102, [97](#)
- sense_struct
 - MAX30102, [80](#)
- sensor, [105](#)
 - _sensor, [111](#)
 - ~sensor, [107](#)
 - averageIRBPM, [111](#)
 - begin, [107](#)
 - BPM_BUFFER_SIZE, [111](#)
 - bpmBuffer, [111](#)
 - Derivative, [107](#)
 - getHR, [108](#)
 - getLatestTemperatureF, [108](#)
 - getPeakThreshold, [108](#)
 - getSpO2, [108](#)
 - HRcalc, [108](#)
 - irLastValue, [111](#)
 - latestIRBPM, [111](#)
 - latestRedBPM, [111](#)
 - latestSpO2, [111](#)
 - latestTemperature, [111](#)
 - localMaximalIR, [111](#)
 - localMaximaRed, [111](#)
 - localMinimalIR, [112](#)
 - localMinimaRed, [112](#)
 - loopThread, [108](#)
 - nextBPMBufferIndex, [112](#)
 - nextSPO2BufferIndex, [112](#)
 - PAST_PEAKS_SIZE, [112](#)
 - pastMaximasIR, [112](#)
 - pastMaximasRed, [112](#)
 - pastMinimasIR, [112](#)
 - pastMinimasRed, [112](#)
 - peakDetect, [109](#)
 - R, [112](#)
 - redLastValue, [113](#)
 - resetCalculations, [109](#)

- runHRCalculationLoop, 109
- running, 113
- runningHR, 113
- sensor, 107
- SPO2_BUFFER_SIZE, 113
- spo2Buffer, 113
- stop, 110
- stopHRcalc, 110
- timeLastIRHeartBeat, 113
- timeLastLoopRan, 113
- timeLastRedHeartBeat, 113
- updateTemperature, 110
- Sensor.cpp
 - crest, 176
 - dataBeenIncreasing, 176
 - hpf, 176
 - lpf, 176
 - nextPastPeaksIndex, 176
 - R, 177
 - SpO2, 177
 - trough, 177
- serialize
 - alert, 17
 - alertPubSubType, 22
- SerializedPayload_t
 - alertPubSubTypes.cxx, 125
- serializeKey
 - alert, 17
- setADCRange
 - MAX30102, 91
- SetdiagnosisRanges
 - diagnosis, 40
- setFIFOAlmostFull
 - MAX30102, 92
- setFIFOAverage
 - MAX30102, 92
- setIterator
 - CircularDelay< type, size >, 26
- setLEDMode
 - MAX30102, 92
- setProximityThreshold
 - MAX30102, 93
- setPROXINTTHRESH
 - MAX30102, 93
- setPulseAmplitudeIR
 - MAX30102, 93
- setPulseAmplitudeProximity
 - MAX30102, 94
- setPulseAmplitudeRed
 - MAX30102, 94
- setPulseWidth
 - MAX30102, 94
- setSampleRate
 - MAX30102, 95
- setup
 - MAX30102, 95
- SHUTDOWN
 - MAX30102.cpp, 171
- shutDown
 - MAX30102, 96
- shutdown.cpp
 - heartSensor, 180
 - main, 180
- size
 - i2c_smbus_ioctl_data, 59
- SLOT_IR_LED
 - MAX30102.cpp, 171
- SLOT_IR_PILOT
 - MAX30102.cpp, 172
- SLOT_NONE
 - MAX30102.cpp, 172
- SLOT_NONE_PILOT
 - MAX30102.cpp, 172
- SLOT_RED_LED
 - MAX30102.cpp, 172
- SLOT_RED_PILOT
 - MAX30102.cpp, 172
- softReset
 - MAX30102, 96
- SpO2
 - Sensor.cpp, 177
- SPO2_BUFFER_SIZE
 - sensor, 113
- spo2Buffer
 - sensor, 113
- SPO2Tracker, 114
 - _s, 117
 - ~SPO2Tracker, 115
 - getVal, 116
 - ping, 116
 - pingThread, 116
 - SPO2Tracker, 115
 - start, 116
 - stop, 117
 - symptomRanges, 117
 - threadRunning, 118
 - tracker, 117
- squareOf
 - DigitalFilters.h, 140
- start
 - diagnosisInterface, 42
 - HRTracker, 55
 - SPO2Tracker, 116
- stdDiagnoses
 - diagnosis, 40
- stdDiagnosis
 - diagnosis, 40
- stop
 - diagnosisInterface, 42
 - HRTracker, 55
 - sensor, 110
 - SPO2Tracker, 117
- stopHRcalc
 - sensor, 110
- STORAGE_SIZE
 - MAX30102.h, 173

- SubListener
 - DeviceSubscriber::SubListener, [119](#)
- subscriber_
 - DeviceSubscriber, [37](#)
- symptom
 - symptomRange, [120](#)
- symptomRange, [119](#)
 - max, [120](#)
 - min, [120](#)
 - symptom, [120](#)
- symptomRanges
 - diagnosisInterface, [42](#)
 - HRTracker, [56](#)
 - SPO2Tracker, [117](#)
- tail
 - MAX30102::Record, [101](#)
- test.cpp
 - heartSensor, [183](#)
 - main, [183](#)
- testSpO2.cpp
 - main, [184](#)
- threadRunning
 - HRTracker, [56](#)
 - SPO2Tracker, [118](#)
- timeLastIRHeartBeat
 - sensor, [113](#)
- timeLastLoopRan
 - sensor, [113](#)
- timeLastRedHeartBeat
 - sensor, [113](#)
- topic_
 - DevicePublisher, [35](#)
 - DeviceSubscriber, [37](#)
- tps, [9](#)
 - pow, [9](#)
- tracker
 - HRTracker, [56](#)
 - SPO2Tracker, [117](#)
- trough
 - Sensor.cpp, [177](#)
- type
 - alertPubSubType, [20](#)
- type_
 - DevicePublisher, [35](#)
 - DeviceSubscriber, [37](#)
- update
 - Differentiator< T >, [44](#)
 - DigitalFilter< Type >, [46](#)
 - HighPassFilter, [48](#)
 - HighPassFilter3, [51](#)
 - LowPassFilter, [65](#)
 - LowPassFilter2, [68](#)
 - LowPassFilter3, [71](#)
 - LowPassFilter3DiffApprox, [74](#)
 - LowPassFilter3MatchedZ, [76](#)
 - MovingAvarageFilter< size >, [99](#)
- updateTemperature
 - sensor, [110](#)
- User
 - User.cpp, [185](#)
- User.cpp
 - addContact, [185](#)
 - contacts, [186](#)
 - name, [186](#)
 - User, [185](#)
- WAKEUP
 - MAX30102.cpp, [172](#)
- wakeUp
 - MAX30102, [97](#)
- word
 - i2c_smbus_data, [58](#)
- writer_
 - DevicePublisher, [35](#)
- x
 - HighPassFilter3, [52](#)
 - LowPassFilter2, [69](#)
 - LowPassFilter3, [72](#)
 - LowPassFilter3DiffApprox, [74](#)
- x1
 - Differentiator< T >, [45](#)
 - HighPassFilter, [49](#)
- xc
 - HighPassFilter3, [52](#)
 - LowPassFilter2, [69](#)
 - LowPassFilter3, [72](#)
 - LowPassFilter3DiffApprox, [74](#)
- y
 - Differentiator< T >, [45](#)
 - HighPassFilter3, [52](#)
 - LowPassFilter2, [69](#)
 - LowPassFilter3, [72](#)
 - LowPassFilter3DiffApprox, [74](#)
 - LowPassFilter3MatchedZ, [77](#)
- y1c
 - HighPassFilter, [49](#)
- yc
 - HighPassFilter3, [52](#)
 - LowPassFilter2, [69](#)
 - LowPassFilter3, [72](#)
 - LowPassFilter3DiffApprox, [75](#)
 - LowPassFilter3MatchedZ, [77](#)