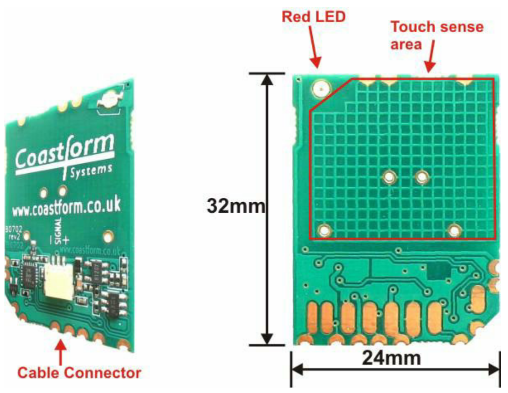
B0702
PROGRAMMABLE CAPACITIVE TOUCH SENSOR/SWITCH



- **Projected capacitive technology (PCAP)**
- **Sense touch through wood, glass, plastic, ceramic etc.**
- **100% auto calibration, for life**
- **Low power**
- **High noise immunity**
- **Advanced controller with robust detect mechanisms**
- **Adjustable sensitivity via software configuration**
- **Programmable output options via software configuration**
- **3-wire connection for power, configuration, and for signaling a remote target**
- **Output transistor for directly switching small relays and other circuits**
- **Serial (UART) interface (on signal wire) for configuration and for communications to other devices such as media players**
- **Free configuration software (Win)**
- **2 metre long cable available**
- **Operating voltage from +5V to +24V DC**
- **Ultra bright LED can be used to acknowledge user's touch**

APPLICATION EXAMPLES -

- LED lighting switch/dimmer
- Hidden switch / proximity sensor
- Temporary switch
- Interactive shop windows
- Digital signage
- Appliances
- Toys
- City information maps
- Shopping centre index
- Vending machines
- Museum exhibit interfaces
- Exhibition interactive displays

These digital, projected-capacitive (PCAP) touch sensor switches detect object approach and object removal as well as human touch. Using the same sensing technology used in the touchscreens of smartphones, they will project touch and sense through almost any non-conductor, such as wood, glass, plastic, stone, ceramic, to a thickness of 3 cm or more, while conductors can be turned into sensors.

The sensor output is a MOSFET transistor that may be configured to operate as a switch, PWM generator or TTL serial transmitter, allowing simple on/off signaling or switching of other devices, speed and intensity control, and advanced control of other electronic devices from media players to PC/Windows applications via binary data sequences and character strings.

A rule based system of events, conditions, actions and timers can be used to program both instant and delayed behaviour that may be different in response to either detect polarity and both start and end of detection.

Configuration information, including sensitivity and the operating behaviour, is stored in an onboard permanent memory and can be changed using Coastform's free QtBtn PC software, together with a simple USB / UART adapter, which also enables detect status and signal levels to be monitored in real time.

These devices are suitable for applications that are subject to environmental influences or even vandalism. They feature adaptive automatic self-calibration, drift compensation, and digital filtering algorithms that make the sensing function robust. They can permit the construction of 100% sealed, watertight sensors and control panels that are immune to humidity, temperature, chemicals, dirt accumulation, abrasion, and abuse.

The sensing area can be increased, if required, by bonding the touch switch to a conductor, such as a small sheet of tin foil, using double-sided adhesive tape.

The board is fitted with a small, cable-capturing, 3-pin socket for both power and signaling. The supply voltage, which should be DC in the range +5V to +24V, uses 2 of the pins while the 3rd is the switching and communications pin.

AVAILABLE OPTIONS

T_A		Part Number	Comments
-10^{0}C to +70^{0}C		B0702-A	Fixed sensitivity. Fixed output function
-10^{0}C to +70^{0}C		B0702-B	Programmable Sensitivity. Programmable output options. Rule based control

1 Overview . . . 4
1.1 Part differences . . . 4
2 Operation . . . 4
2.1 Signal Levels . . . 4
2.2 Detection Integrator . . . 4
2.3 Startup and Normal Mode . . . 4
2.4 Wiring . . . 5
Table 2.1 - Connector Pin List . . . 5
3 Communications . . . 6
3.1 Transistor (FET) Output . . . 6
3.2 PWM Output . . . 6
3.3 UART Communications . . . 6
3.3.1 Configuration Mode . . . 6
3.3.1.1 Reading Data from the Device . . . 7
3.3.1.2 Writing Data to the Device . . . 7
4 Memory Map . . . 8
5 Configuration . . . 9
5.1 Detect Threshold - PTHR, NTHR . . . 9
5.2 Drift Compensation - PDRIFT, NDRIFT . . . 10
5.3 Detect Integrators - PDIL, NDIL . . . 10
5.4 Recal Delay - PRD, NRD . . . 11
5.5 Burst Length - BL . . . 11
5.6 Hysteresis - PHYST, NHYST . . . 12
5.7 Sleep - SLEEP . . . 12
5.8 Time to Enter Configuration Mode - TEC . . . 13
5.9 Time Device Remains Awake After Detect Events - AWAKE . . . 13
5.10 Configuration CRC - CCRC . . . 13
7 Rules - Events, Conditions, Actions, Timers . . . 14
7.1 Event Rules . . . 14
7.1.1 Rule Type 1 - Event: Initialisation End . . . 15
7.1.2 Rule Type 2 - Event: Positive Detect . . . 15
7.1.3 Rule Type 3 - Event: Positive Release . . . 15
7.1.4 Rule Type 4 - Event: Negative Detect . . . 15
7.1.5 Rule Type 5 - Event: Negative Release . . . 15
7.1.6 Rule Type 6 - Event: Enter Normal Mode . . . 15
7.1.7 Rule Types 10 to 17 - Event: Timer Elapse, High-Resolution, Short-Interval Timers 0 to 7 . . . 15
7.18 Rule Types 18 to 21 - Event: Timer Elapse, Low Resolution, Long Interval Timers 8 to 11 . . . 16
7.2 Condition Rules . . . 17
7.2.1 Rule Type 50 - Condition: Detect . . . 17
7.2.2 Rule Type 51 - Condition: Timer . . . 17
Timer Types:Hi-Res,Short = High-Resolution, Short-IntervalLo-Res,Long = Low-Resolution, Long-Interval . . . 18
7.2.3 Rule Type 52 - Condition: Variable 1 is Greater Than . . . 18
7.2.4 Rule Type 53 - Condition: Variable 1 is Less Than . . . 18
7.2.5 Rule Type 54 - Condition: Variable 2 is Greater Than . . . 18
7.2.6 Rule Type 55 - Condition: Variable 2 is Less Than . . . 18
7.2.7 Rule Type 56 - Condition: Duty Cycle Setting is Greater Than . . . 19
7.2.8 Rule Type 57 - Condition: Duty Cycle Setting is Less Than . . . 19
7.2.9 Rule Type 58 - Condition: On/Off . . . 19
7.3 Action Rules . . . 20
7.3.1 Rule Type 100 - Action: Output Drive . . . 20
7.3.2 Rule Type 101 - Action: Transmit Byte . . . 21
7.3.3 Rule Type 102 - Action: Set PWM Duty Cycle . . . 21
7.3.4 Rule Type 103 - Action: Start or Cancel Timer . . . 21

Timer Types:Hi-Res,Short = High-Resolution, Short-IntervalLo-Res,Long = Low-Resolution, Long-Interval 22
7.3.5 Rule Type 104 - Action: Set Variable 1 22
7.3.6 Rule Type 105 - Action: Set Variable 2 22
7.3.7 Rule Type 106 - Action: Change Variable 22
7.3.8 Rule Type 107 - Action: LED ON/OFF 23
7.4 Rule Timers 24
Timer Types:Hi-Res,Short = High-Resolution, Short-IntervalLo-Res,Long = Low-Resolution, Long-Interval 24
8 Specifications 25
8.1 Absolute Maximum Electrical Specifications 25
8.2 Recommended operating conditions 25
Appendix A - CRC Algorithm 27
A.1 16-Bit CRC Software C Algorithm 27

1 Overview

ProxiTouch devices are digital burst mode charge-transfer (QT) sensors designed specifically for touch controls and object sensing; they include all signal processing functions necessary to provide stable sensing under a wide variety of changing and challenging conditions.

ProxiTouch parts either employ self capacitance sensing, a technology that senses changes in charge acquired from a sensor electrode charged and discharged by pulse edges, or transverse charge-transfer ('QT') sensing, a technology that senses changes in electrical charge forced across an electrode by a pulse edge.

1.1 Part differences

Versions of the sensor switch are available with either fixed function or programmable behaviour. These units are physically identical, but differ in their capabilities.

B0702-A has a factory programmed configuration that cannot be changed, including a simple output switch function. The sensitivity is fixed and set quite high so that, for example, it can be operated through a 12mm thick glass window. The output transistor is switched on during touch and switched off when the touch is removed.

B0702-B has many configurable operating parameters, including sensitivity, has a programmable rule system of events, conditions, actions and timers, and has programmable output behaviour, supporting PWM compatible output drive, serial data transmissions for targets such as media players as well as a simple switch output.

2 Operation

The sensor is sampled in a burst of acquisition pulses, the number of which are determined by the configuration parameter BL. At the end of each burst the resulting signal is converted to digital form and processed. The burst length directly affects touch gain.

2.1 Signal Levels

Using Coastform's QtBtn software it is easy to observe the absolute level of signal received by the sensor on each sample. QtBtn software is available free of charge on Coastform's website.

Immediately after power-up or reset, the device calibrates to establish a reference level, against which the signal is compared after each future sample. A detect is declared when the signal deviates from this reference by more than the threshold a number of times. To ensure robust and reliable operation, the reference value is drift compensated at a slow rate over time.

The signal swing resulting from events such as the smallest finger touch or object removal should preferably exceed 10 counts. The threshold setting (PTHR & NTHR) should be set to a value guaranteed to be less than the signal swing caused by events to be detected.

Increasing the burst length (BL) parameter will increase the signal strengths.

The device can be configured to detect signal swings in either direction. That is, greater than the reference or less than the reference. An approaching finger causes the signal to rise, or swing in the positive direction, so the signal has a greater valu e than the reference, whereas removing an object from the close vicinity of the sensor causes the signal to drop, or swing in the negative direction, so the signal is less than the reference.

The sensor may be configured to detect either positive or negative signal swings, or both. Configuration parameters for positive detects are PTHR, PDIL, and PHYST. Configuration parameters for negative detects are NTHR, NDIL, and NHYST. Separate events and conditions are available to process positive and negative detects.

2.2 Detection Integrator

The devices feature a detection integration mechanism, which acts to confirm a detection in a robust fashion. A counter is incremented each time the sensor signal has exceeded its threshold and stayed there for a number of acquisitions. When this counter reaches a preset limit a detect is finally declared. Example: if the limit value is 10, then the sensor has to exceed its threshold and stay there for a minimum of 10 acquisitions before detect is declared.

The detection integrator operates equally for signal swings of either direction. PDIL defines the counter limit for positive signal swings and NDIL defines the limit for negative swings.

2.3 Startup and Normal Mode

B0702-B has two main modes, normal mode and configuration mode. Immediately after power-up / reset it goes through a short initialisation period and then enters a short time window, when configuration mode can be entered, before automatically transitioning to normal mode if it has not entered configuration mode.

Some output actions, such as switching the output on, will take effect only when the device is in normal mode, and are ignored in configuration mode and during initialisation and during the the time window to enter configuration mode. This behaviour is necessary to prevent these output actions from otherwise corrupting configuration communications or preventing the device from entering configuration mode.

2.4 Wiring

Table 2.1 - Connector Pin List
Applies to all units

Pin	Function		Comments	Cable Colour
1	--	P	0V, ground	Black
2	SIGNAL	I/O	Signal output, Configuration wire.	Brown
3	+	P	+5V to +24V DC supply	Red
P = Power. I/O = Input/Output				

3 Communications

A transistor (FET) is used at the output of these devices, enabling them to directly drive other equipment. The B0702-B is also equipped with a TTL-level UART, allowing it to be configured and for it to communicate simple commands to target equipment.

The B0702-B can be configured to generate a pulse width modulated (PWM) waveform at its output, via the output transistor which can be used to drive other electronics including dimming LEDs.

The B0702-B supports 3 different output modes, transistor drive (on or off), PWM output, and UART communications, but only one can be used at a time because all the output modes share the same circuitry, including the output transistor and the SIGNAL pin on the connector.

The B0702-B will accept UART communications, for configuration and monitoring, only when it is in configuration mode. and there is a short time window to enter configuration mode immediately after power-up.

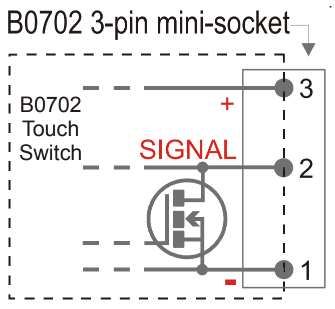
3.1 Transistor (FET) Output

Both devices operate a0 transistor (FET) at the output (**Figure 3-1**), connected across the SIGNAL and 0V pins, which can sink up to 50mA and drive LEDs, relays, micro-controller inputs and other circuits.

The B0702-A has a fixed output drive behaviour, turning the transistor on while a touch is detected, and turning if off again when the touch is removed.

The B0702-B transistor behaviour is completely flexible and must be defined by the Rules (see section 7 Rules - Events, Conditions, Actions, Timers).

Figure 3-1 Signal Output / Communication Pin



3.2 PWM Output

0B0702-B can be configured to generate a PWM signal, with configurable duty cycle. This output behaviour is useful for dimming LEDs as well as other circuits. The PWM output status (on/off) and duty cycle are fully configurable via the Rules (see section 7 Rules - Events, Conditions, Actions, Timers).

3.3 UART Communications

UART communications are used for two purposes on the B0702-B, for configuration and signal monitoring, and to send command sequences to computers, media players and other devices to alert them of detect events, for example, and otherwise control them.

The B0702-B uses a single wire, named SIGNAL, for bi-direction half-duplex UART communications. This enables the use of a compact 3-pin connector.

The baud rate is fixed.

SIGNAL - Transmit/Receive asynchronous data. The output transistor is used as a transmitter driver and is arranged in an open drain circuit as shown in **Figure 3-1** Note that this circuit arrangement can only generate a logic low level itself. To generate a logic high level, an off-board pull-up resistor is required, pulled-up to the voltage the external equipment operates the bus at (see section 6 Specifications).

UART transmission parameters are:

Baud rate:	57,600
Start bits:	1
Data bits:	8
Parity:	None
Stop bits:	1

3.3.1 Configuration Mode

Configuration mode uses the built-in TTL-level UART to enable the device to be programmed with touch sensor configuration and rules for the device behaviour as well as to enable the signals to be monitored. This mode can be entered during a short time window immediately after power-up by sending the device a specific sequence of three bytes. The B0702-B enables its UART receiver on power-up and monitors the SIGNAL line during this time window. Configuration mode is entered upon receipt of the following sequence of bytes : 0x01, 0x80, 0x01, which is the communication sequence to read back the device's revision.

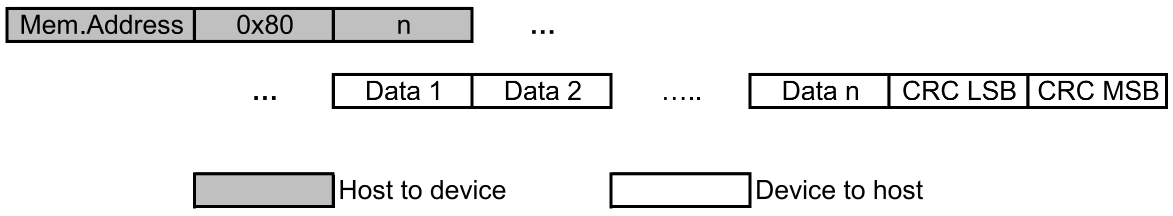
If the narrow time window expires without configuration mode being entered, the UART receiver is disabled and the device enters normal operation, or normal mode. The width of the time window may be set using the TEC configuration parameter.

The device automatically exits configuration mode, and transitions to normal mode, if there is a continuous period of 10 seconds of inactivity at the SIGNAL line. Essentially, this is 10s after a monitoring host such as QtBtn ceases communication.

In configuration mode, the B0702-B supports both read and write operations on a memory mapped interface.

3.3.1.1 Reading Data from the Device

The data sequence to read data from the device is shown below



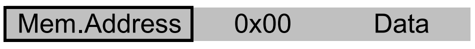
Key to read sequence

Text	Description
Mem.Address	Target memory address within device
0x80	Read bit. Indicates to the device this is a read operation.
n	# bytes to read. Valid range is 1 - 253. CRC is available after reading n bytes.
Data	Data from device.

The host sends 3 command bytes, indicating the address it wishes to start the read from, a read bit, and the number of bytes it wishes to read, then waits for the device to return the requested data. The host initiates the transfer by sending the internal memory address it wishes to read from. It then sends a byte with value 0x80 which tells the B0702-B this is a read operation. It then sends n, where n is a count of the number of data bytes it wishes to read. The device transmits n data bytes. Each time a data byte is transmitted, the device automatically increments the internal address so that subsequent data bytes are read from consecutive addresses. When all n data bytes have been returned, the device returns a 16-bit CRC, LSB first.
The device calculates the 16-bit CRC using the 3 command bytes transmitted to the device (the internal memory start address, the 0x80, the number of bytes to be read, n), and the n data bytes themselves, all in the same sequence they occur during the transmission (see Appendix A).

3.3.1.2 Writing Data to the Device

The data sequence to write data to the device is shown below



The host sends just 3 bytes in each sequence, the target memory address it wishes to write, a second byte with value 0x00, and the byte of data to be written. Only one location in the device can be written in each sequence; A separate write sequence must be executed for each byte to be written to the device.

4 Memory Map

The following table shows the memory map.

Table 4.1 Memory Map

Address	Use	Access
0	Device ID (35)	Read
1	Revision	Read
2	Reserved	Read
3	Reserved	Read
4	Device Status, as a collection of bit flags	Read
5	Detect Status, for positive detect	Read
6	Detect Status, for negative detect	Read
7 - 8	Sensor Signal	Read
9 - 10	Sensor Reference	Read
11	DI count	Read
12	Reserved	Read
13	Reserved	Read
14	Reserved	Read
15 - 20	Reserved	Read
21	Control command address. Write 0xFF to calibrate the sensor. Write 0xFE immediately before writing configuration. Write 0x01 to reset the device.	Write
22 - 27	Sensor Configuration	Read/Write
28 - 35	Rule Timers, Hi-Res, Short-Interval	Read/Write
36 - 43	Rule Timers, Lo-Res, Long-Interval	Read/Write
44 - 253	Rule Table	Read/Write
254 - 255	Configuration CRC, Host defined	Read/Write

Reading/writing the appropriate locations allows the data there to be read back or changed. Locations that report the sensor signal and other runtime operating parameters are read only. Attempts to write to these locations are ignored. Locations marked 'reserved' may contain random data.

All data, including the configuration parameters are accessible via the memory mapped interface while the device is in configuration mode only.

Where a value takes up more than one location the data order is little endian.

5 Configuration

The device configuration is stored in internal non-volatile EEPROM memory, and is accessible through the appropriate locations in the memory map only when the device is in configuration mode. It is broken down into subsections which permit configuration of the touch sensor, the rule timers, the rule table, and an optional CRC. The touch sensor configuration parameters are discussed here, the rule timers and rules are discussed in section 7.

The devices calibrate and process all signals using a number of algorithms specifically designed to provide for high survivability in the face of adverse environmental challenges. They provide a large number of processing options which can be user-selected to implement very flexible, robust touch solutions.

Because the signal deviates from the reference in both the positive and negati ve directions, many configuration parameters are divided into two sub-components, one to support positive signal deviations and the other for negative deviations, with th e parameter names being prefixed with either P or N respectively.

Its possible to configure the device to detect positive signal movements, where the signal moves to a higher value than the reference, negative movements, where the signal is less than the reference, or both. A positive signal deviation typically occurs when a finger or hand approaches the sensor and is therefore useful for touch detection, whereas a negative deviation usually occurs when the sensor is moved away from a nea rby object and can therefore be used to sense object removal.

User-defined configuration are employed to alter these algorithms to suit each application. The parameters are loaded into the device over the TTL-level UART interface when the device is in configuration mode. **After a setups change, the power should be cycled to ensure the device has calibrated using the latest settings**.

Many configuration parameters employ lookup table (LUT) translation.

Default Values shown are factory defaults. The configuration of the B0702-A is programmed in the factory and cannot be changed.

5.1 Detect Threshold - PTHR, NTHR

PTHR, NTHR

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
27	PTHR	PTHR	PTHR	PTHR	NTHR	NTHR	NTHR	NTHR

The threshold value is established relative to the sensor's reference value, and is used to determine detect when crossed by a either a positive or negative-going signal swing after having been filtered by the detection integrator. Larger absolute values of threshold desensitize the sensor since the signal must travel farther in order to cross the threshold level. Conversely, lower thresholds make the sensor more sensitive.

The amount of threshold required depends on the amount of signal swing that occurs during a detect. Thicker touch surfaces or smaller touch geometries reduce sensitivity, thus requiring smaller values of PTHR or NTHR to achieve detect.

PTHR sets the threshold for a signal greater than the reference, whereas NTHR sets the threshold used when the signal drops below the reference.

Detect Threshold - Lookup Table

PTHR, NTHR Index	Threshold
0	10
1	12
2	16
3	22
4	30
5	40
6	52
7	66
8	82
9	100
10	120
11	142
12	166
13	192
14	220
15	254

The lookup table (LUT) shown on the left is used to translate the 16 PTHR or NTHR indices into the required threshold. The value to program to the device configuration is the LUT index, not the actual threshold.

5.2 Drift Compensation - PDRIFT, NDRIFT

PDRIFT, NDRIFT

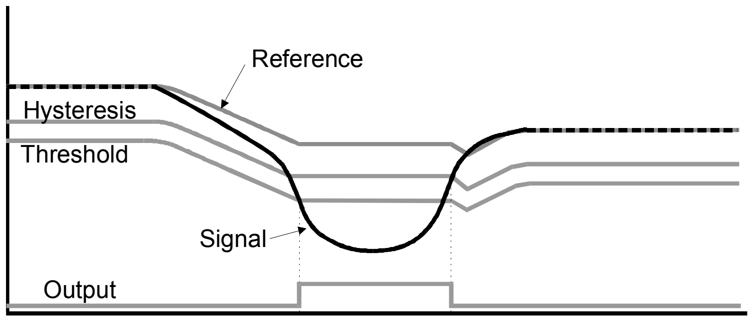
Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
22	PDRIFT	PDRIFT	PDRIFT	BL	BL	NDRIFT	NDRIFT	NDRIFT

Signals can drift because of changes in electrical properties over time and temperature. It is crucial that such drift be compensated or else false detections and sensitivity shifts can occur.

Drift compensation (**Figure 5-1**) is performed by making the reference level track the raw signal at a slow rate, but only while there is no detect in effect. The rate of adjustment must be performed slowly, otherwise legitimate detections could be ignored. The devices drift compensate using a slew-rate limited change to the reference level, with the threshold and hysteresis values slaved to this reference.

When a finger or other object is sensed, the signal rises since nearby objects act to increase the capacitance of the sensor. Once a detect is in effect, the drift compensation mechanism ceases since the signal is legitimately detecting an object. Drift compensation only works when the signal has not crossed one or other threshold level.

Figure 5-1 Thresholds and Drift Compensation



The drift compensation mechanism can be made asymmetric, if desired, so it operates in one direction faster than it does in the other simply by changing the PDRIFT and NDRIFT configuration parameters.

The drift compensation should not be set to operate too quickly since an approaching finger could be compensated for partially or entirely before even touching the touch pad.

Drift compensation and the detection time-outs work together to provide for robust, adaptive sensing. The time-outs provide abrupt changes in reference calibration depending on the duration of the signal 'event'.

Drift Compensation - Lookup Table

PDRIFT, NDRIFT Index	Drift Interval (s)
0	0.2
1	0.4
2	0.8
3	1.2
4	2
5	3.3
6	6
7	10

The lookup table (LUT) shown on the left is used to translate the 8 PDRIFT or NDRIFT indices into the required drift compensation interval, in seconds. The value to program to the device configuration is the LUT index, not the actual interval.

5.3 Detect Integrators - PDIL, NDIL

PDIL, NDIL

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
23	PDIL	PDIL	PDIL	PDIL	NDIL	NDIL	NDIL	NDIL

To suppress false detections caused by spurious events like electrical noise, the device incorporates a 'detection integrator' or DI counter mechanism that acts to confirm a detection by consensus (enough detect events must agree). The DI mechanism counts detect events, of the signal crossing the threshold, after each sample. For a detect to be declared, the DI counter must reach the user-set level; Only when the DI counter reaches PDIL or NDIL does a detect become formally 'active'.

The DI is extremely effective at reducing false detects, but at the expense of slower reaction times. In some applications a slow reaction time is desirable; the DI can be used to intentionally slow down touch response in order to require the user to hold a touch longer to operate the switch.

When **Signal > Reference +Positive-Threshold** : The DI counter is incremented towards the target of PDIL .

When **Signal < Reference -Negative-Threshold** : The DI counter is incremented towards the target of NDIL .

Detect Integrators- Lookup Table

PDIL, NDIL Index	Detect Integrator Limit
0	2
1	3
2	5
3	7
4	9
5	12
6	15
7	20
8	32
9	45
10	60
11	90
12	123
13	150
14	190
15	254

The lookup table (LUT) shown on the left is used to translate the 16 PDIL or NDIL indices into the required detect integrator limit. The value to program to the device configuration is the LUT index, not the actual limit.

5.4 Recal Delay - PRD, NRD

PRD, NRD

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
24	PRD	PRD	PRD	PRD	NRD	NRD	NRD	NRD

If an object is moved near or away from a sensor resulting in a detection for a prolonged interval it is usually desirable to re-calibrate the sensor to cancel the detect and restore normal operation, perhaps after a time delay of some seconds.

The Recal Delay timer monitors such detections; If a detection event exceeds the timer's setting, the touch will be automatically canceled and the reference re-calibrated. After a re-calibration has taken place, the sensor will once again function normally even in the continued presence or absence of the foreign object. This feature is set using PRD and NRD. PRD sets the time-out when a positive detect (signal > reference) is active and NRD sets the time-out when a negative detect (signal < reference) has occurred.

PRD and NRD cannot be disabled.

Recal Delay - Lookup Table

PRD, NRD Index	Recal Delay (s)
0	2
1	3
2	5
3	7
4	9
5	12
6	15
7	20
8	32
9	45
10	60
11	90
12	123
13	150
14	190
15	254

The lookup table (LUT) shown on the left is used to translate the 16 PRD or NRD indices into the required re-calibration delay, in seconds. The value to program to the device configuration is the LUT index, not the actual delay.

5.5 Burst Length - BL

BL

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
22	PDRIFT	PDRIFT	PDRIFT	BL	BL	NDRIFT	NDRIFT	NDRIFT

The signal gain is controlled largely by the burst length.

The burst length is simply the number of times the charge-transfer ('QT') process is performed to accumulate charge before each sample is taken, and may be selected from one of four different built-in values.

Increasing burst length directly affects touch sensitivity because the accumulation of charge in the charge integrator is directly linked to the burst length.

Apparent touch sensitivity is also controlled by the threshold level (PTHR and NTHR) resulting in an interaction between burst length and threshold. The burst length should be kept short to prevent overloading the charge accumulator, but threshold should be as high as possible to reduce false detections due to external noise. The detection integrator mechanism also helps to prevent false detections.

Setting up burst length and threshold can be a fine balance. Many combinations might provide a working solution but one may offer the best noise immunity. The recommended procedure is to start with a short burst length and set the threshold so that the sensor operates reliably, as required, but without over-sensitivity. If the sensitivity appears rather poor, increase the burst length and adjust the threshold again. Beware of setting the burst length too long as this can result in overloading the charge accumulator and is counter productive. If the charge accumulator is overloaded, sensitivity will suffer and the sensor may not detect at all. Observe the signal deviation (using Coastform's QtBtn) during detect events with different burst lengths and if the deviation is only marginally better, or worse, with a longer burst length setting the sensor is overloaded and the burst length should be shortened.

Burst Length - Lookup Table

BL Index	Burst Length (pulses)
0	2
1	4
2	8
3	16

The lookup table (LUT) shown on the left is used to translate the 4 BL indices into the required number of pulses that comprise the burst length. The value to program to the device configuration is the LUT index, not the actual burst length.

5.6 Hysteresis - PHYST, NHYST

PHYST, NHYST

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
25	PHYST	PHYST	NHYST	NHYST	TEC	SLEEP	SLEEP	SLEEP

The devices employ programmable hysteresis levels of 6.25%, 12.5%, 25%, or 50%. The hysteresis is a percentage of the distance from the threshold level back towards the reference, and defines the point at which a detection will drop out. A 12.5% hysteresis point is closer to the threshold level than to the reference level.

Hysteresis prevents chatter and works to make detection more robust. Hysteresis is used only once the detect has been formally declared, in order to determined when the detect should drop out.

Excessively large amounts of hysteresis can result in a stuck detect that does not release after touch, for example, especially when signal deviation levels are small. Low amounts of hysteresis can cause detect chatter due to signal noise or minor amounts of finger motion.

Hysteresis - Lookup Table

PHYST, NHYST Index	Hysteresis (%)
0	6.25
1	12.5
2	25
3	50

The lookup table (LUT) shown on the left is used to translate the 4 PHYST or NHYST indices into the required hysteresis. The value to program to the device configuration is the LUT index, not the actual hysteresis

5.7 Sleep - SLEEP

PHYST, NHYST

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
25	PHYST	PHYST	NHYST	NHYST	TEC	SLEEP	SLEEP	SLEEP

The device may be optionally configured to sleep between samples. Sleep may be disabled (SLEEP = 0), or may be set to one of seven different intervals, in which case the device will sleep for the selected interval between samples, unless other conditions prevent it, minimising power consumption. Longer sleep intervals yield progressively lower power consumption, but at the expense of response time.

Any of the following conditions can temporarily prevent the sleep interval between samples : configuration mode is active, during the short time window after power-up when configuration mode may be entered, PWM output is in use, any rule timer is running, the awake timeout is active (see AWAKE), the device is calibrating, detect is declared, the signal deviation exceeds the threshold in either direction.

CAUTION - Do not enable sleep if the voltage on the SIGNAL pin can exceed 3.5V at any time, otherwise permanent damage could result to the B0702-B. If the voltage on SIGNAL can ever exceed 3.5V, disable sleep using SLEEP = 0.

Sleep - Lookup Table

SLEEP Index	Sleep Interval (ms)
0	Sleep disabled
1	16
2	32
3	64
4	125
5	250
6	500
7	1,000

The lookup table (LUT) shown on the left is used to translate the 8 PDRIFT or NDRIFT indices into the required drift compensation interval, in seconds. The value to program to the device configuration is the LUT index, not the actual interval.

5.8 Time to Enter Configuration Mode - TEC

PHYST, NHYST

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
25	PHYST	PHYST	NHYST	NHYST	TEC	SLEEP	SLEEP	SLEEP

Configuration mode can be entered during a narrow time window immediately after power-up. The width of this time window may be set to one of two values with TEC.

It can be more difficult to enter configuration mode when the window is narrow. Convers ely, some applications may require touch sensing to be operational as soon as possible after power-up. User's may therefore select the wider TEC window while testing various configuration parameters and rules and repeatedly switching in and out of configuration mode, and elect to use the narrow window thereafter, during normal operation.

The wider window is the default setting.

Time to Enter Configuration Mode- Lookup Table

TEC Index	Window Width (s)
0	1
1	10

The lookup table (LUT) shown on the left is used to translate the 2 TEC indices into the time for the window. The value to program to the device configuration is the LUT index, not the actual time.

5.9 Time Device Remains Awake After Detect Events - AWAKE

AWAKE

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
26	AWAKE	AWAKE	AWAKE	AWAKE	AWAKE	AWAKE	AWAKE	AWAKE

Some applications may require very low power consumption and therefore demand a long sleep interval between samples. However, the lower power consumption comes at the expense of longer response time, and whilst a longer response time may be an accep table after a long period of touch inactivity, it may be desirable to have a faster response while the touch switch is in use. The AWAKE configuration caters for this scenario. The device will not sleep between samples within a time interval, measured from the end of any detect, the duration of which is set by AWAKE.

The awake time-out has a range from 0.1s to 25.5s and is calculated as follows:

Awake Time-out = AWAKE * 0.1s

where AWAKE is an 8-bit integer in the range 1 - 255. Zero is an invalid value.

The AWAKE parameter is effective only if sleep is enabled.

5.10 Configuration CRC - CCRC

CCRC

Address	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
254 , 255	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC	CCRC

The configuration area terminates with a 16-bit CRC, which can be used by a host to store a CRC calculated on the configuration area. The use of this area is defined by the host.

7 Rules - Events, Conditions, Actions, Timers

The B0702-B may be programmed using a simple but flexible structure of rules broken down into the three categories: events, conditions and actions. These rules allow control over what the touch switch does in response to touch and other events. For example, the switch can be programmed to operate its output transistor when touched, or to operate as an LED lighting dimmer, or transmit the character sequence "ON" via the signal pin when touched. The possibilities are huge.

The rules are programmed into a rule table with sufficient space for a maximum of 70 rules. The first entry in the table has an index of zero, and the last entry has an index of 69. Each entry consists of three bytes, byte 0, byte 1, and byte 2.

Rule Table

Rule Index #	Note	Rule Byte 0	Rule Byte 1	Rule Byte 2
0	First rule	Rule type	Defined by rule type	
1				
...				
68				
69	Last rule			

Byte 0 -Defines the rule category and type. The category is one of Event, Condition or Action, and is implied by the type value. The rule type value must be in the range 0 to 255, although not all of these values are defined; Some are reserved for future expansion. Only those Rule types defined by current Event, Condition and Action types are listed in this document. Unused/reserved Rule types are not listed.

Byte 1 -The meaning is defined by the rule category and type.

Byte 2 -The meaning is defined by the rule category and type.

7.1 Event Rules

The programmable touch switch supports a number of different event types, including touch, object removal, and a rule timer elapsing.

After the occurrence of each event, the B0702-B checks the rule table for matching events, processing each one as it is found. If multiple matching events are found they are all processed, one at a time, in order. Event rule checking starts at the beginning of the rule table (Rule 0), continues in ascending order of index within the table, and stops either when a void rule is found or the end of the rule table is reached, whichever occurs fist; A void Rule is one where Byte0 has the value 255.

Order of event rule processing within rule table

Index #		Processing order	
0	First rule	First rule checked by event engine	V
1			V
...		Event rule processing is aborted if a void rule (Byte0 = 255) is found.	V
			V
68			V
69	Last rule	Last rule checked by event engine	

Event rules can be located throughout the rule table, but for optimum processing efficiency it is recommended to create them as a single contiguous block starting at Rule 0, and terminated with a void rule. If the event rules are scattered throughout the rule table, or are not terminated with a void rule, the rule engine will have to skip over other rules(conditions & actions) to find all the event rules.

Only matching events are processed. All condition rules, all action rules and all non-matching event rules are skipped during the search for matching event rules.

Each event rule must include an event type and can optionally include conditions and normally also includes an action, as shown in the table below.

Use of bytes within event rules

Byte #	Description
0	Event Rule type
1	Index of first Condition Rule to apply to this Event
2	Index of first Action Rule to execute for this Event

The action determines what the switch does in response to the event, and the condition(s) allow the action to be conditionally invoked, if the conditions are met, when the event occurs.

Byte 0 -Defines the event type, which must be one of those listed below.

Byte 1 -Condition. An index to a condition rule within the table. A void condition index of 255, or an index beyond the end of the rule table, indicates that no conditions apply to this event rule. If a valid index is specified, the event will continue to be processed only if the condition(s) are met. If the conditions are not met the action will not be invoked. A valid index must reference a real condition rule within the table, otherwise the behaviour is undefined.
Multiple conditions can be associated with an event rule, see the section on conditions for details.

Byte 2 -Action. An index to an action rule within the table. A void action index of 255, or an index beyond the end of the rule table, indicates that no actions apply to this event rule. A valid index must reference a real action rule within the table, otherwise the behaviour is undefined.
Multiple actions can be associated with an event rule, see the section on actions for details.

7.1.1 Rule Type 1 - Event: Initialisation End

The Initialisation event occurs after power-up when the touch switch internal initialisation has completed and the time window to enter configuration mode has begun.

This event can be used to automatically trigger an action or sequence of actions after the touch switch is powered on or reset. For example, the onboard LED could be configured to flash once per second.

Some care is needed when using this event because some output actions, such as switching the output on, are inhibited during the time window to enter configuration mode.

7.1.2 Rule Type 2 - Event: Positive Detect

A positive detect. occurs when the B0702-B is touched.

7.1.3 Rule Type 3 - Event: Positive Release

A Positive Release event occurs after a touch is removed.

7.1.4 Rule Type 4 - Event: Negative Detect

Negative detect typically occurs when something is moved away from the touch switch. The negative detect event could be used, for example, to detect an object removed from a stand in a museum; The touch sensor might be a metal sheet located in or on the top of the stand.

7.1.5 Rule Type 5 - Event: Negative Release

A Negative Release event occurs when a negative detect condition ends, either because the removed object is returned or because the touch switch re-calibrates.

7.1.6 Rule Type 6 - Event: Enter Normal Mode

This event occurs when the touch switch enters normal mode which, in turn, occurs either at the end of the short time window when configuration mode can be entered, or upon exit from configuration mode.

This event can be useful when testing a new set of Rules. For example, the LED may be configured to flash once when the switch enters normal mode, perhaps to help the tester know when new rules, specifically with actions that control the output, can start to be tested.

7.1.7 Rule Types 10 to 17 - Event: Timer Elapse, High-Resolution, Short-Interval Timers 0 to 7

There are eight high-resolution, short-interval timers. After a rule timer has been started it will run for the preset interval, at the end of which an elapse event will occur with a rule type specific to the elapsed rule timer.

Rule Type versus rule timer #

Rule Type	Index # of Elapsed Hi-Res Timer
10	0
11	1
12	2
13	3
14	4
15	5
16	6
17	7

The correct rule type must be specified in a rule to match a specific rule timer interval elapse event.

The rule timer is canceled upon the interval elapsing. it does not automatically restart another interval.

7.18 Rule Types 18 to 21 - Event: Timer Elapse, Low Resolution, Long Interval Timers 8 to 11

There are four low-resolution, long-interval timers. After a rule timer has been started it will run for the preset interval, at the end of which an elapse event will occur with a rule type specific to the elapsed rule timer.

Rule Type versus rule timer #

Rule Type	Index # of Elapsed Lo-Res Timer
18	8
19	9
20	10
21	11

The correct rule type must be specified in a rule to match a specific rule timer interval elapse event.

The rule timer is canceled upon the interval elapsing. it does not automatically restart another interval.

7.2 Condition Rules

Conditions may be applied to events so that the corresponding actions are invoked only when all the conditions are met at the time of the event. One or more conditions may apply to each event type, and a number of different condition types are supported.

Use of bytes within condition rules

Condition Rule Byte #	Description
0	Condition rule type
1	Other information specific to the condition
2	Index of next condition rule that also applies with this condition rule. 255 = no further conditions.

Byte 0 -Defines the condition type, which must be one of those listed below.

Byte 1 -This data is specific to each condition rule. See each condition rule for details of how this byte is used.

Byte 2 -Link to another condition. An index to another condition rule within the table. A void index of 255, or an index beyond the end of the rule table, indicate there are no further conditions. A valid index must reference a real condition rule within the table, otherwise the behaviour is undefined.

A number of conditions may be applied to an event by linking, or daisy chaining, conditions together using byte 2 in each condition. Where multiple conditions are linked together, all the conditions must be satisfied when an event occurs for the corresponding actions to be invoked. The actions will not be invoked if any condition is not met.

7.2.1 Rule Type 50 - Condition: Detect

This condition can be used to check if either of the detect types is confirmed.

Byte 1 of the condition rule indicates the detect condition to test for.

Byte 1 value	Condition Tested
0	No detect
1	Positive detect is active
2	Negative detect is active

7.2.2 Rule Type 51 - Condition: Timer

This Condition can be used to check if a rule timer interval is running or is stopped. A rule timer is considered to be running if it is within its countdown interval. A rule timer interval that has never been started, or previously been running and either elapsed or has been canceled will test as stopped.

Byte 1 of the condition rule indicates the rule timer to check and the rule timer state to check.

Byte 1 Use

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Running / Not Running	-	-	-	Timer #			

Bit 7 - Indicates whether to test for the rule timer running or stopped. 0 = test rule timer is stopped. 1 = test timer is running.
Bits 6..4 - Reserved.
Bits 3..0 - A 4-bit integer that indicates the rule timer number to test. Valid values are in the range 0 to 11. Other values are invalid and should not be used.

The following table shows all valid values for Byte 1, and the corresponding condition tested.

Byte 1 values to test if a rule timer is running or not

Byte 1 value	Bit 7 value	Timer #	**Condition Tested**	Rule Timer Type
0	0	0	Timer 0 stopped	Hi-Res,Short
1	0	1	Timer 1 stopped	Hi-Res,Short
2	0	2	Timer 2 stopped	Hi-Res,Short
3	0	3	Timer 3 stopped	Hi-Res,Short
4	0	4	Timer 4 stopped	Hi-Res,Short
5	0	5	Timer 5 stopped	Hi-Res,Short
6	0	6	Timer 6 stopped	Hi-Res,Short
7	0	7	Timer 7 stopped	Hi-Res,Short
8	0	8	Timer 8 stopped	Lo-Res,Long
9	0	9	Timer 9 stopped	Lo-Res,Long
10	0	10	Timer 10 stopped	Lo-Res,Long
11	0	11	Timer 11 stopped	Lo-Res,Long
128	1	0	Timer 0 running	Hi-Res,Short
129	1	1	Timer 1 running	Hi-Res,Short
130	1	2	Timer 2 running	Hi-Res,Short
131	1	3	Timer 3 running	Hi-Res,Short
132	1	4	Timer 4 running	Hi-Res,Short
133	1	5	Timer 5 running	Hi-Res,Short
134	1	6	Timer 6 running	Hi-Res,Short
135	1	7	Timer 7 running	Hi-Res,Short
136	1	8	Timer 8 running	Lo-Res,Long
137	1	9	Timer 9 running	Lo-Res,Long
138	1	10	Timer 10 running	Lo-Res,Long
139	1	11	Timer 11 running	Lo-Res,Long
Timer Types: Hi-Res,Short = High-Resolution, Short-Interval Lo-Res,Long = Low-Resolution, Long-Interval				

7.2.3 Rule Type 52 - Condition: Variable 1 is Greater Than
There are two internal variables, variable 1 and variable 2, which are under the total control of the programmed rules and maybe used within the rules to conditionally change behaviour.

This condition may be used to test if Variable 1 is greater in value than the value in Byte 1.

7.2.4 Rule Type 53 - Condition: Variable 1 is Less Than
There are two internal variables, variable 1 and variable 2, which are under the total control of the programmed rules and maybe used within the rules to conditionally change behaviour.

This condition may be used to test if Variable 1 is less than the value in Byte 1.

7.2.5 Rule Type 54 - Condition: Variable 2 is Greater Than
There are two internal variables, variable 1 and variable 2, which are under the total control of the programmed rules and maybe used within the rules to conditionally change behaviour.

This condition may be used to test if Variable 2 is greater in value than the value in Byte 1.

7.2.6 Rule Type 55 - Condition: Variable 2 is Less Than
There are two internal variables, variable 1 and variable 2, which are under the total control of the programmed rules and maybe used within the rules to conditionally change behaviour.

This condition may be used to test if Variable 2 is less than the value in Byte 1.

7.2.7 Rule Type 56 - Condition: Duty Cycle Setting is Greater Than

The B0702-B has a PWM generator that can be configured with a duty cycle setting in the range 0 to 255. This condition may be used to test if the duty cycle setting is greater than the value in Byte 1.

7.2.8 Rule Type 57 - Condition: Duty Cycle Setting is Less Than

The B0702-B has a PWM generator that can be configured with a duty cycle setting in the range 0 to 255. This condition may be used to test if the duty cycle setting is less than the value in Byte 1.

7.2.9 Rule Type 58 - Condition: On/Off

Condition On/Off allows the on/off state of items to be tested.

Byte 1 value	Condition Tested
0	PWM generator output = OFF
1	PWM generator output = ON
2	Onboard LED = OFF
3	Onboard LED = ON
4	Output = OFF
5	Output = ON. This corresponds to the output transistor on.

All other values for Byte 1 not listed in the above table are invalid and should not be used.

7.3 Action Rules

Action rules are used to define what the touch switch does in response to events. For example, to switch on the on-board LED, to switch off the output transistor, to generate a PWM waveform, to transmit a string of characters.

One or more actions may be executed for each event rule. More than one action can be specified by linking actions together in a daisy chain fashion, using Byte 2 of each action rule to specify the index of the next action rule in the chain.

Use of bytes within action rules

Byte #	Description
0	Action rule type
1	Other information specific to the action
2	Next action rule that should also be executed. 255 = no further actions to execute.

Byte 0 -Defines the action type, which must be one of those listed below.

Byte 1 -This data is specific to each action rule. See each action rule for details of the meaning.

Byte 2 -Link to another action. An index to another action rule within the table. Where multiple actions are associated with an event rule, they are executed in the same order they are listed in the chain, starting with the action referenced in the event rule.
A void index of 255, or an index beyond the end of the rule table, indicates there are no further actions. A valid index must reference a real action rule within the table, otherwise the behaviour is undefined.

Actions that change the state at the output transistor are not executed if the B0702-B is not in normal mode. This is to prevent the action interfering with the communications between the B0702-B and the configuration host, such as QtBtn. If an action is inhibited in this manner, any other action rules linked to it are unaffected and will continue to be processed.

7.3.1 Rule Type 100 - Action: Output Drive

Byte 1 of the action rule indicates the type of output to generate.

Byte 1 value	Description of Action
0	**Set output OFF.** Set the output transistor to a permanently off state
1	**Set output ON.** Set the output transistor to a permanently on state
2	**Toggle output.** Toggle the state of the output transistor on/off
3	**Set PWM generator OFF.** The PWM duty cycle is unaffected by switching the generator off. The setting for the output transistor becomes effective, no longer being overridden by the PWM generator.
4	**Set PWM generator ON.** The PWM generator will operate with the current duty cycle setting. This setting overrides the setting for the output transistor. That is, the PWM generator will be effective regardless of the setting for the output transistor.
5	**Toggle PWM generator on/off.** If the generator is off, it is switched on and this will override the setting for the output transistor. If the generator is on, it is switched off and the setting for the output transistor becomes effective. The duty cycle setting is unaffected.
6	**Transmit the value from Variable 1.** The value from Variable 1 is transmitted as serial data through the output transistor, over the signal pin, by the internal UART. The output transistor should be switched off before issuing this action. A pull-up resistor is required at the input to the receiver.
7	**Transmit the value from Variable 2.** The value from Variable 2 is transmitted as serial data through the output transistor, over the signal pin, by the internal UART. The output transistor should be switched off before issuing this action. A pull-up resistor is required at the input to the receiver.
8	**Transmit the PWM duty cycle setting.** The current duty cycle setting is transmitted as serial data through the output transistor, over the signal pin, by the internal UART. The output transistor should be switched off before issuing this action. A pull-up resistor is required at the input to the receiver. The PWM generator should be switched off before issuing this action.

9	**Enter configuration mode.** The PWM generator is switched off, and the output transistor is switched off before the device enters configuration mode. The device will remain in configuration mode indefinitely until the power is cycled,or a UART character start-bit is detected on the signal pin. The internal configuration mode time-out interval is started as soon as a start-bit is detected, and configuration mode is exited when the time-out interval elapses.

Only those Byte 1 values listed in the above table are valid. All other values are ireserved and should not be used. This action will not be executed if the B0702-B is not in normal mode.

7.3.2 Rule Type 101 - Action: Transmit Byte

The value from Byte 1 of the action rule is transmitted as serial data through the output transistor, over the signal pin, by the internal UART.
The output transistor should be switched off before issueing this action. A pull-up resistor is required at the input to the receiver.

This action will not be executed if the B0702-B is not in normal mode.

7.3.3 Rule Type 102 - Action: Set PWM Duty Cycle

Set the PWM duty cycle using the value defined in Byte 1. The active state of the PWM generator is unchanged, if the generator is off it remains off. If it is on, operation continues with the new duty cycle.

The value from Byte 1 of the action rule determines the duty cycle, D, as defined by

$$D = (\text{Byte } 1 / 255) \times 100\%$$

Range of values 0 - 255 (0 - 100%).

This action will not be executed if the B0702-B is not in normal mode.

7.3.4 Rule Type 103 - Action: Start or Cancel Timer

This action can be used to start or cancel a rule timer. If a timer is started it will run for the full preset interval, unless it is subsequently cancelled, and is automatically stopped at the end of the interval. Thus these timers can be termed 'one-shot'. If a repeating timer interval is required, appropriate rules must be configured to restart the timer at the end of each interval, in response to the timer expired event.

Byte 1 of this Rule indicates the rule timer and whether to start or cancel it.

Byte 1 Use							
Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
Start / Cancel	-	-	-	Timer #			

Bit 7 - Indicates whether to start or cancel the rule timer. 0 = cancel. 1 = start.
Bits 6..4 - Reserved.
Bits 3..0 - A 4-bit integer that indicates the rule timer number. Valid values are in the range 0 to 11. Other values are invalid and should not be used.

The following table shows all valid values for Byte 1, and the corresponding action.

Byte 1 values to start or cancel a rule timer

Byte 1 value	Bit 7 value	Timer #	**Action**	Rule Timer Type
0	0	0	Cancel Timer 0	Hi-Res,Short
1	0	1	Cancel Timer 1	Hi-Res,Short
2	0	2	Cancel Timer 2	Hi-Res,Short
3	0	3	Cancel Timer 3	Hi-Res,Short
4	0	4	Cancel Timer 4	Hi-Res,Short
5	0	5	Cancel Timer 5	Hi-Res,Short
6	0	6	Cancel Timer 6	Hi-Res,Short
7	0	7	Cancel Timer 7	Hi-Res,Short
8	0	8	Cancel Timer 8	Lo-Res,Long
9	0	9	Cancel Timer 9	Lo-Res,Long
10	0	10	Cancel Timer 10	Lo-Res,Long
11	0	11	Cancel Timer 11	Lo-Res,Long
128	1	0	Start Timer 0	Hi-Res,Short
129	1	1	Start Timer 1	Hi-Res,Short
130	1	2	Start Timer 2	Hi-Res,Short
131	1	3	Start Timer 3	Hi-Res,Short
132	1	4	Start Timer 4	Hi-Res,Short
133	1	5	Start Timer 5	Hi-Res,Short
134	1	6	Start Timer 6	Hi-Res,Short
135	1	7	Start Timer 7	Hi-Res,Short
136	1	8	Start Timer 8	Lo-Res,Long
137	1	9	Start Timer 9	Lo-Res,Long
138	1	10	Start Timer 10	Lo-Res,Long
139	1	11	Start Timer 11	Lo-Res,Long
Timer Types: Hi-Res,Short = High-Resolution, Short-Interval Lo-Res,Long = Low-Resolution, Long-Interval				

Only those Byte 1 values shown in the above table are valid. All other values are reserved and should not be used.

A rule timer with an active time-out interval will report as running. A timer that is canceled has its time-out interval reset to zero and reports as stopped. A timer that has never been started also reports as stopped.

See also the section on timers.

7.3.5 Rule Type 104 - Action: Set Variable 1
Set internal Variable 1 to the value in rule Byte 1.

7.3.6 Rule Type 105 - Action: Set Variable 2
Set internal Variable 2 to the value in Rule Byte 1.

7.3.7 Rule Type 106 - Action: Change Variable
This action supports various operations on Variable 1, Variable 2, and the PWM duty cycle.

Byte 1 of the action rule indicates the action to perform.

Byte 1 value	**Description of Action**
0	**Decrement Variable 1.** Result: Variable1 = Variable1 -1 The operation does not attempt to be well behaved at boundaries; If Variable 1 is zero beforehand, the operation will cause it to wrap around the byte boundary and the result will be 255.
1	**Increment Variable 1.** Result: Variable1 = Variable1 +1 The operation does not attempt to be well behaved at boundaries; If Variable 1 is 255 beforehand, the operation will cause it to wrap around the byte boundary and the result will be 0.
2	**Logical Shift Right, Variable 1 (Divide Variable 1 by 2).** Operation: 0→b7→b6→b5→b4→b3→b2→b1→b0→ Result: Variable1 = Variable1 /2 This is a crude method to divide by 2. After the operation the result is always an even integer.
3	**Logical Shift Left, Variable 1 (Multiply Variable 1 by 2).** Operation: ←b7←b6←b5←b4←b3←b2←b1←b0← 0 Result: Variable1 = Variable1 x2 The operation does not attempt to be well behaved at boundaries; If bit 7 is set (Variable 1 > 127) beforehand, the operation will cause that bit to be lost and the value will be reduced.

4	**Decrement Variable 2.** Result: Variable2 = Variable2 -1 The operation does not attempt to be well behaved at boundaries; If Variable 2 is zero beforehand, the operation will cause it to wrap around the byte boundary and the result will be 255.
5	**Increment Variable 2.** Result: Variable2 = Variable2 +1 The operation does not attempt to be well behaved at boundaries; If Variable 2 is 255 beforehand, the operation will cause it to wrap around the byte boundary and the result will be 0.
6	**Logical Shift Right, Variable 2 (Divide Variable 2 by 2).** Operation: 0→b7→b6→b5→b4→b3→b2→b1→b0→ Result: Variable2 = Variable2 /2 This is a crude method to divide by 2. After the operation the result is always an even integer.
7	**Logical Shift Left, Variable 2 (Multiply Variable 2 by 2).** Operation: ←b7←b6←b5←b4←b3←b2←b1←b0← 0 Result: Variable2 = Variable2 x2 The operation does not attempt to be well behaved at boundaries; If bit 7 is set (Variable 2 > 127) beforehand, the operation will cause that bit to be lost and the value will be reduced.
8	**Decrement Duty Cycle Setting.** Result: PDC = PDC -1 The operation does not attempt to be well behaved at boundaries; If PDC is zero beforehand, the operation will cause it to wrap around the byte boundary and the result will be 255.
9	**Increment Duty Cycle Setting.** Result: PDC = PDC +1 The operation does not attempt to be well behaved at boundaries; If PDC is 255 beforehand, the operation will cause it to wrap around the byte boundary and the result will be 0.
10	**Logical Shift Right, Duty Cycle Setting (Divide PDC by 2).** Operation: 0→b7→b6→b5→b4→b3→b2→b1→b0→ Result: PDC = PDC /2 This is a crude method to divide by 2. After the operation the result is always an even integer.
11	**Logical Shift Left, Duty Cycle Setting (Multiply PDC by 2).** Operation: ←b7←b6←b5←b4←b3←b2←b1←b0← 0 Result: PDC = PDC x2 The operation does not attempt to be well behaved at boundaries; If bit 7 is set (Variable 1 > 127) beforehand, the operation will cause that bit to be lost and the value will be reduced.
12	**Copy Variable 2 to Variable 1.** Result: Variable1 = Variable2
13	**Copy Variable 1 to Variable 2.** Result: Variable2 = Variable1
14	**Copy Duty Cycle Setting to Variable 1.** Result: Variable1 = PDC
15	**Copy Variable 1 to Duty Cycle Setting.** Result: PDC = Variable1
16	**Add Variable 2 to Variable 1.** Result: Variable1 = Variable1 +Variable2 The operation does not attempt to be well behaved at boundaries; If the result would be greater than 255, it will actually be truncated.
17	**Subtract Variable 2 from Variable 1.** Result: Variable1 = Variable1 - Variable2 The operation does not attempt to be well behaved at boundaries; If Variable 2 is greater than Variable 1 beforehand, the result is undefined.

PDC = PWM Duty Cycle setting.

Only those Byte 1 values shown in the above table are valid. All other values are reserved and should not be used.

7.3.8 Rule Type 107 - Action: LED ON/OFF

This action is used to set the onboard LED on, or off or toggle its state.

Byte 1 of the action rule indicates the action to perform.

Byte 1 value	Description of Action
0	Set LED OFF
1	Set LED ON
2	Toggle LED on/off. If the LED is on, this action will turn it off. If it is off, it will be switched on.

Only those Byte 1 values shown in the above table are valid. All other values are reserved and should not be used.

7.4 Rule Timers

The B0702-B has twelve timers, called rule timers, that can be programmed to work in conjunction with the Events,Conditions & Actions Rules. For example, two timers could be programmed, one with a short interval of perhaps 100ms, and the other with a longer interval of 2s so that together with a few rules, the onboard LED can be configured to deliver a quick flash every 2 seconds. A more complicated example might allow an LED lighting dimmer mechanism to be programmed with various delays to support quick touches to switch the LED lights on or off, and touch-and-hold, to dim the lights etc.

The timers are divided into two groups, one with high-resolution but short interval, and the other with low-resolution but longer maximum interval.

Rule Timers

Timer Index #	Rule Timer Type	Resolution	Maximum Interval
0	Hi-Res,Short	10ms	2550ms (>2.5s)
1	Hi-Res,Short	10ms	2550ms (>2.5s)
2	Hi-Res,Short	10ms	2550ms (>2.5s)
3	Hi-Res,Short	10ms	2550ms (>2.5s)
4	Hi-Res,Short	10ms	2550ms (>2.5s)
5	Hi-Res,Short	10ms	2550ms (>2.5s)
6	Hi-Res,Short	10ms	2550ms (>2.5s)
7	Hi-Res,Short	10ms	2550ms (>2.5s)
8	Lo-Res,Long	2s	> 36 hours
9	Lo-Res,Long	2s	> 36 hours
10	Lo-Res,Long	2s	> 36 hours
11	Lo-Res,Long	2s	> 36 hours
Timer Types: Hi-Res,Short = High-Resolution, Short-Interval Lo-Res,Long = Low-Resolution, Long-Interval			

The first eight timers (Timers 0 to 7) have a high resolution of 10ms and a maximum interval of 2550ms, or greater than 2.5 seconds, in 255 equal 10ms steps. An internal lookup table (LUT) is used to translate the 255 indices into the required time interval. The value programmed to the rule timer is the LUT index, not the actual time interval.

Timers 8 to 11 support intervals up to 131,070s, or more than 36 hours, with a 2s resolution. They use a 16-bit, or 65,535 index, lookup table (LUT) to map the index into the required time interval. The value programmed to the rule timer is the LUT index, not the actual time interval.

Timers 0 - 7 (High-Resulution, Short Interval)

Resolution:	10ms
Range:	1 - 255(10ms - 2550ms, in 10ms steps). Zero is invalid, do not use.
Time Interval: =	LUT-index x 10ms
Accuracy:	Single shot: +1ms/-10ms, Repeating: +/-1ms

Timers 8 - 11 (Low-Resolution, Long Interval)

Resolution:	2s
Range:	1 - 65535(2s - 131,070s, in 2s steps). Zero is invalid, do not use.
Time Interval: =	LUT-index x 2s
Accuracy:	+/-1s

8 Specifications

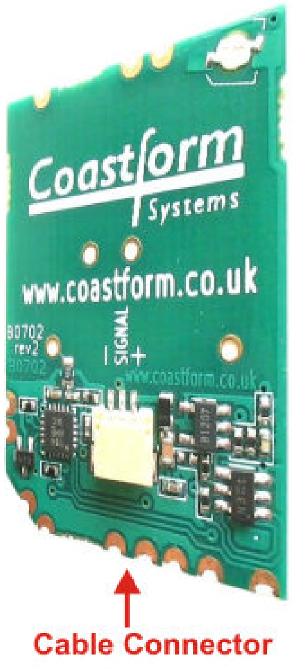
8.1 Absolute Maximum Electrical Specifications

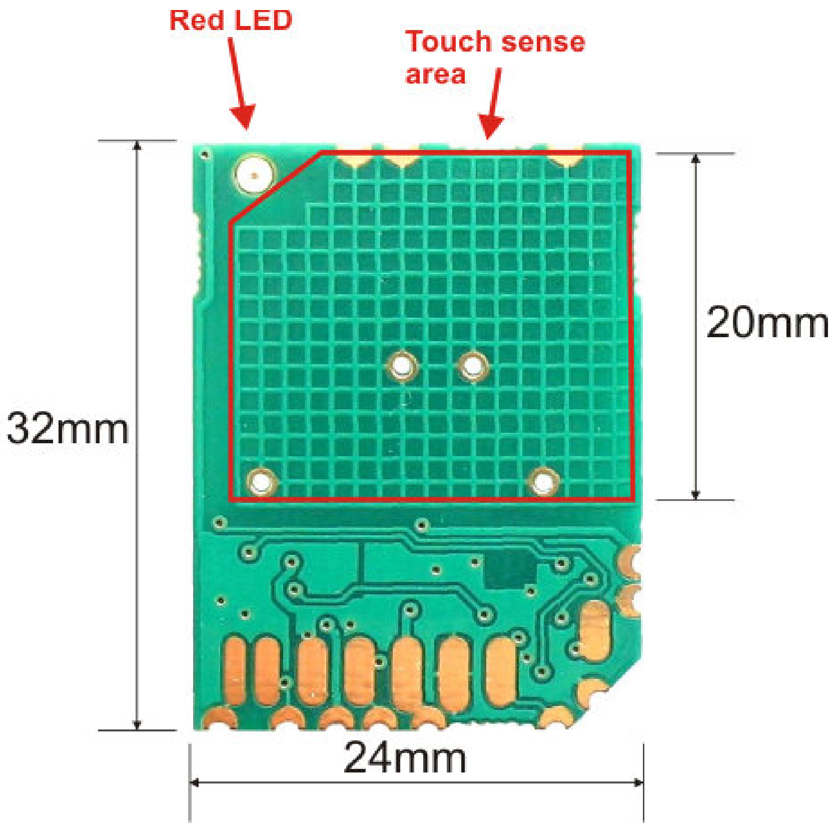
Operating temp.	-10^{O}C ~ +70^{O}C
Storage temp.	-55^{O}C ~ +85^{O}C
Supply Voltage.	+5V ~ +24.5V
Voltage on SIGNAL output line (sleep disabled).	+30V
Voltage on SIGNAL output line (sleep enabled).	+3.5V
Max continuous output current (sink), at up to +30V DC	-50mA
Configuration maximum writes (B0702-B only).	100,000 write cycles

8.2 Recommended operating conditions

V_{DD}.	+5V ~ +24V

7.4 Mechanical Dimensions





Appendix A - CRC Algorithm

A.1 16-Bit CRC Software C Algorithm

```
/*
16 bits crc calculation.
Initial crc entry value must be 0. The message is not augmented with 'zero' bits.
Polynomial = X16 + X15 + X2 + 1
data is an 8 bit number, unsigned (uint8_t)
crc is a 16 bit number, unsigned (uint16_t)
Repeat this function for each byte, folding the result back into the call parameter, crc
*/
uint16_t Crc_16Bit(uint16_t wCRC, uint8_t cData)
{
        uint8_t c;        /* loop counter */

        wCRC = wCRC ^ ((uint16_t)cData << 8);

        c = 8;
        do
        {
                if(wCRC & 0x8000)
                {
                        wCRC = (wCRC << 1) ^ 0x1021;
                }
                else
                {
                        wCRC = (wCRC << 1);
                }
                c--;
        } while(c);

        return(wCRC);
}
```

Phoenix House, Rotheram Road
Dinnington, Sheffield, S25 3RG, UK
Tel: +44 (0)1909 561470

sales@coastform.co.uk
www.coastform.com

Revisions

To 1.00
Initial copy

1.01
Major update, information expanded in most sections.

1.02
Major update, added & corrected most information on the QT sensor configuration.

1.03
Added text regarding normal and config. modes.
For clarity, changed the name of Rule Type 6 - Event: from "Exit Configuration Mode" to "Enter Normal Mode".

1.04
Changed text on cover page.