



Summary

DARC-I is a standalone Data Acquisition and Remote Control (DARC) module. It is controlled by a remote device which sends it commands via a Bluetooth link.

Hardware Features

- FCC / CE / IC certified Class I Bluetooth V1.1 radio, 100m range, integral antenna.
- 12 analog inputs (10-bit).
- 5 PWM outputs (10-bit).
- 7 digital dedicated digital I/O pins.
- All analog and PWM pins may be also configured for digital I/O.
- Digital I/O pins can be configured as one 7-bit and/or up to two 5-bit parallel digital I/O.
- 64K flash, 2.3K RAM and 1K EEPROM memory available on-board.
- Up to 256K I2C external memory.
- Real time clock.
- Onboard power regulator, 5V – 10V supply.



phone and module not to scale

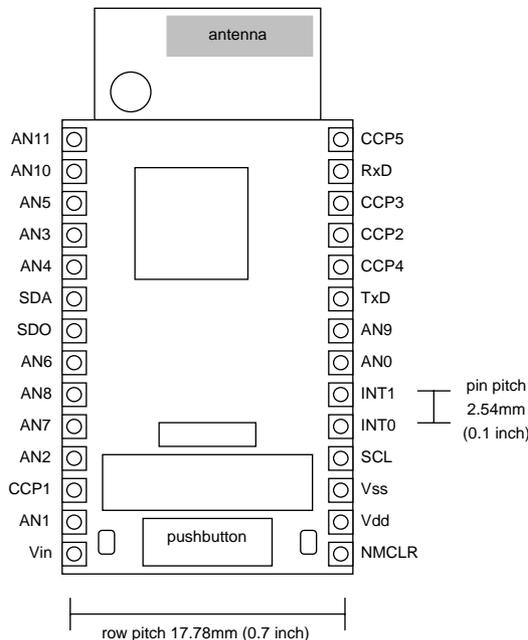
Firmware Features

DARC-I has a message-based command language allowing it to be configured via a Bluetooth link. The command set includes:

- PWM, digital and parallel output control.
- Analog, digital and parallel input control.
- Streaming of input data direct to remote device.
- Capture of frames of input data up to 5K samples / sec.
- Long-term data logging based on real time clock, even with remote device unconnected.
- Daylight savings time management.
- Bluetooth security and settings.

Customization

- Firmware C source code and customization services available.
- Customization possibilities include faster sampling rates and power saving modes.



Manufactured to ISO9001:2000



Ordering Information

Part No	Description
	DARC-I Standard 28-pin Dual-in-Line package

Contents

SUMMARY	1
CONTENTS	2
PIN DESCRIPTIONS	2
EVALUATING DARC-I	3
COMMANDS	5
Reset Command	6
Configure DARC-I Command	6
Configure I/O Command	7
Set I/O Command	7
Get I/O Command	7
Get Memory Command	7
Set Memory Command	7
Stream Data Command	7
Frame Data Command	7
Logging Data	7
RESPONSES	7
OK Response	7
Error Response	7
Date Time Response	7
I/O Value Response	7
Got Memory Response	7
Initialization Response	7
COMMAND / RESPONSE USER GUIDE	7
CUSTOMIZATION	7
MECHANICAL DATA	7
TECHNICAL SPECIFICATIONS	7
Electrical	7
Radio	7
FCC, CE and IC modular approval	7
ORDERING CONTACT DETAILS	7
TECHNICAL SUPPORT AND CUSTOMIZATION	7

Pin Descriptions

Pin Names	Description
AN0-AN11	Analog input or digital I/O (see note 2)
CCP1-CCP5	PWM output or digital I/O (see note 2)
SDA, SCL	External memory interface or digital I/O (see note 2)
SDO, TxD, RxD, INT0, INT1	Digital I/O (see note 2)
NMCLR	50ms pulse low to reset. May be left unconnected.
Vdd	Regulated power +5V (see note 1,2)
Vin	Unregulated power input +5 to +10V (see note 1)
Vss	Power ground reference

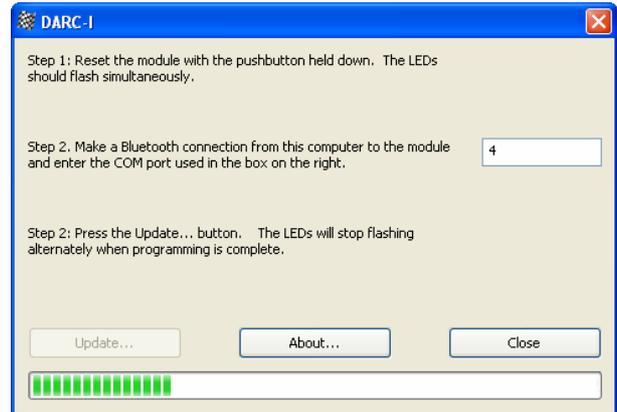
1. Either (i) regulated power should be provided on Vdd and Vin left unconnected or (ii) unregulated power should be provided on Vin and Vdd may be used as a regulated power output.
2. If on-board power regulator used, total current draw on all outputs (including Vdd if used as a regulated power output) shall not exceed 130mA.

Evaluating DARC-I

DARC-I is based upon the ToothPIC module from FlexiPanel Ltd. The Evaluation Version is intended for OEMs to use to evaluate the technology prior to customizing to their product lines.

The DARC-I is supplied as a ToothPIC module which must be 'Field Programmed'. This takes a few seconds and requires either a Windows PC or a Pocket PC with Bluetooth. The procedure is as follows. If required use the default PIN code "0000".

1. Download the ToothPIC Development Kit from www.flexipanel.com and locate the DARC-I Service Pack Darc-I Win.exe (Windows) or Darc-I PPC.exe (Pocket PC).
2. Power-up the ToothPIC with the on-board pushbutton held down. The on-board LEDs will flash simultaneously.
3. Start running the DARC-I Service Pack and connect from the computer to the ToothPIC using Bluetooth.
4. Enter the COM port used to connect to the ToothPIC in the box provided.
5. Press the Update button. Programming takes about 30 seconds. When the progress bar is full, field programming is complete.



DARC-I is intended to respond to binary commands, but it can also accept the same commands if typed in ASCII hexadecimal. To send typed ASCII hexadecimal commands to DARC-I, you will need to connect it to a terminal emulator. The instructions which follow assume you will use the HyperTerminal program which comes bundled free with the Windows operating system. You will need a Windows PC with Bluetooth. You must also complete all steps each time, since some steps rely on preceding steps being complete.

6. When the module powers up, the two LEDs flash once to show that the module has initialized correctly. It then enters a discoverable and connectable state. From your Windows PC, search for the DARC-1 module and connect to it. Note which COM port Bluetooth uses to make the connection. The start HyperTerminal and select the following settings:
 - Connect using COM port as appropriate.
 - Specify 9600 baud, 8 data bits, no parity, 1 stop bit, hardware flow control.
7. As soon as HyperTerminal connects, the following message should appear:

```
11444152432D4920332E302E3030303034
```

This is the hex codes for the number 17 followed by the 17 text characters DARC-I 3.0.00004. (The version number may differ.) Go to *File > Properties > Settings > ASCII Setup* in the menu and set the following:

- Check *Send line ends with line feeds*.
- Check *Echo characters typed locally*.

8. Type the following command. If you make a mistake, simply tap 'z' until you get a 0302F1 ('not understood') error message and then start again.

```
04033101
```

You should get the response 0201, meaning command completed OK. You sent the command to turn on the red LED and it should have come on. The first byte of the command, 04, indicates that the whole command is 4 bytes long. The 03 indicates that the command is to set an I/O output value. The 32 indicates the I/O value is the red LED and finally 01 indicates you want to turn it on. You can set all outputs this way.

9. To configure I/O as outputs or inputs, the 02 command is used. Type the following to set AN0 and AN1 as analog inputs:

```
04020102
```

10. To read an input, the 04 command is used. Type the following to read the AN0 analog input:

```
030410
```

You should get the response 050410xxyy, meaning AN0 was measured to be xxyy, where xx is the low byte and yy high byte. The exact value will depend on the stray voltages present on the pins.

There are three types of repetitive data recording available:

Streaming allows data to be sent at regular intervals direct to the host. The speed is limited to the Bluetooth communications rate (approx 5K – 10K bytes / sec). Streaming continues until the host tells DARC-I to stop.

Framing allows a fixed amount of data to be recorded in memory at regular intervals. The speed is only limited by the speed at which the inputs can be sampled. The host can upload the data after all the data has been collected. This allows DARC-I to capture snapshots of high-speed data, for example for spectral analysis.

Logging allows data to be recorded in memory at regular intervals as dictated by the real time clock. The difference between framing and logging is that you must wait for the framing command to complete before doing anything else. In logging, you can send other commands while logging is taking place. You can even disconnect and reconnect later. The host can upload the data when it reconnects.

11. To request logged data, send the following commands:

```
0501231001          (asks for AN0 to be recorded; expect a 0201 response)
```

```
0501231101          (asks for AN1 to be recorded; expect a 0201 response)
```

```
0501240202          (asks to store samples in RAM)
```

```
04012201            (log samples once per second)
```

Data are being logged once per second. The samples are stored in RAM, as specified. If the RAM fills up, logging will start from the beginning of RAM again, overwriting the oldest data. To stop logging, use the command:

```
04012200            (log samples once per second)
```

The data are stored as an 8-byte date-time field followed by the sampled data, which in this case is two two-byte analog samples. To retrieve samples, use Get Memory commands. For example, type:

06050200000C

(get first sample's 12 bytes, i.e. date-time, AN0 & AN1)

The response I received while writing this guide was:

110502000003020B010601D00700001402

The 03020B meant the time was 12:02:03. The 010601D007 meant the clock date was Saturday (0x06), 1st January, 2000 (0x07D0). The samples were 0x0000 (AN0) and 0x00214 (AN1).

12. To request framed data, send the following commands. (You must have done step 10 first to select AN0 and AN1 for recording, and note that this command turns off the real time clock so you cannot log data again without resetting.)

0501210100

(sample rate 5000 samples per second)

04080002

(store a frame of 512 values)

After the final command, 512 samples are logged to memory at a rate of 5000 samples per second and then the '0201' acknowledge command is transmitted. The Get Memory command is used to retrieve the data, the same as for logging, although there is no date-time field now, e.g.

060502000010

(get first 16 bytes, e.g. first 4 samples)

13. To request streamed data, send the following commands:

0501217D00

(sample rate 4 samples per second)

0207

(starts data streaming)

You will get a four lots of four hexadecimal bytes message `xyyaabb` every second. AN0 was measured to be `xyyy`. AN1 was measured to be `aabb`. Send any character to end the data streaming.

Commands

Commands can be in either ASCII or binary and the two formats can be mixed freely. In ASCII format, each byte is transmitted as a two hexadecimal digits (upper or lower case) and the entire command must be followed by a <CR><LF> pair (i.e. the control characters 0x0D and 0x0A). DARC-I can differentiate between ASCII and binary commands by looking at the first character. If it is less than the ASCII character '0' (zero), the command is binary. If it is an ASCII character for a hexadecimal digit, the command is ASCII. Any other character is an illegal command.

Commands may be up to 47 bytes long. This The first byte is the *command length byte*, equal to the total number of bytes in the message. The second byte is the *command byte*, which indicates how the remainder of the message should be interpreted.

Only one command can be processed at once. To know when the previous command has completed, observe the state of the RTS pin or wait for a response to be sent. Only then send another command.

Command Summary		
Command	Command Byte	Effect
Reset	0x00	Disconnects and resets
Configure DARC-I	0x01	Configures DARC-I
Configure I/O	0x02	Configures I/O
Set I/O	0x03	Sets an I/O value

Command Summary		
Command	Command Byte	Effect
Get I/O	0x04	Requests an I/O value
Read Memory	0x05	Reads from memory locations
Write Memory	0x06	Writes to memory locations
Stream Data	0x07	Streams data samples to remote device
Get Data Frame	0x08	Stores a sequence of data samples in memory
Log Data	n/a	

Reset Command

The command byte 0x00 instructs the ToothPIC to disconnect and reset.

Reset Command Examples	
Reset (binary)	0x02 0x00
Reset (ASCII)	"0200<CR><LF>"

Configure DARC-I Command

The command byte 0x01 configures the general properties of the DARC-I. The byte after the command byte is the *Property Byte*, which specifies the exact property being set. The remaining bytes represent the new property value, as follows:

Configure DARC-I Command Properties		
Property Byte	Property	Remaining Byte(s)
0x01	Security level*	00 = None (default) 01 = Authentication 02 = Authentication and encryption
0x02	Authentication PIN*	Zero terminated ASCII pin code (maximum 16 characters plus zero terminator)
0x03	Device name*	Zero terminated ASCII device name (maximum 16 characters plus zero terminator)
0x04	Initialization response*	00 = No an initialization response FF = Generate an initialization response (default)
0x05	ASCII responses*	00 = Generates binary responses FF = Generates ASCII responses (default)
0x06	On internal error...*	01 = Flash error number, reset on button press 02 = Reset immediately 03 = Send error response to host (default)
0x07	I2C memory setup*	00 = No I2C memory (default) 01 = I2C memory with 100kHz clock speed 02 = I2C memory with 400kHz clock speed
0x08	Device class*	3-byte device class value as defined below
0x21	Set stream / frame rate	2-byte <i>SampleRate</i> (low byte first). Sets the sample rate to multiples of 200µS.
0x22	Set log rate	1-byte <i>LogRate</i> value as defined below

Configure DARC-I Command Properties		
Property Byte	Property	Remaining Byte(s)
0x23	Set stream / frame / log inputs	1st byte: Input ID value as described in Stream Data Command 2nd byte: 00 = Don't sample 01 = Sample
0x24	Set frame / log memory locations	2-bytes <i>FLMemStart</i> , <i>FLMemEnd</i> as defined below
0x41	Daylight Savings	1-byte <i>DSTEvent</i> value as defined below
0x42	Date / Time	8-byte value as defined below
0x43	Request Date / Time	No additional bytes. Generates a Date Time response immediately.

Items marked * are stored permanently in Flash memory. With the exception of the PIN code, these changes will not take effect until after the device next resets. If required, the default values can be restored by reloading the DARC-I Firmware Solution using Wireless Field Programming as described in steps 1-5.

LogRate determines when data is logged to memory:

LogRate	Definition
0x00	No logging. (Streaming / framing permitted.)
0x01	Every second
0x02	Every 2 seconds
0x03	Every 5 seconds
0x04	Every 10 seconds
0x05	Every 15 seconds
0x06	Every 30 seconds
0x07	Every 60 seconds
0x08	Every 2 minutes
0x09	Every 5 minutes
0x0A	Every 10 minutes
0x0B	Every 15 minutes
0x0C	Every 30 minutes
0x0D	Every 60 minutes
0x0E	Every 2 hours
0x0F	Every 3 hours
0x10	Every 6 hours
0x40...0x57	Daily at midnight / 1 am / 2 am / ... / 11 pm

The Set stream / frame / log inputs property includes or excludes an input from the set of data to be recorded. It must also be configured as an appropriate input.

FLMemStart, *FLMemEnd* determine where data is stored. Each of *FLMemStart* and *FLMemEnd* are *mStr* memory types as defined for the Set Memory command. It will start at the first allowable location in *FLMemStart* and work along linearly until no more records will fit. It will then move to the next memory type and record the whole of the next record at the beginning of that; when memory *FLMemEnd* is full, it will restart at *FLMemStart*.

For example, if both *FLMemStart* and *FLMemEnd* are 0x01 (RAM), it will fill the RAM memory from 0x000 to 0x8FF and then start from 0x000 again. If *FLMemStart* is 0x14 (External memory 0xB8) and *FLMemEnd* is 0x81 (ROM), it will fill external memory 0xB8, then external memory 0xBA then external memory 0xBC, then external memory 0xBE, then the Flash ROM, and then start back with external memory 0xB8. All external memory will be assumed to contain valid addresses 0x0000 to 0x7FFF, i.e. 24C256 and 24C512 compatible devices.

DSTEvent defines the next moment in time at which the Real Time Clock will adjust itself. When it does so, it also swaps the Daylight Savings Time value for its seasonal converse.

<i>DSTEvent Value (decimal)</i>	<i>Definition</i>	<i>Month</i>	<i>Date</i>	<i>Day of week</i>	<i>Hour</i>	<i>Action taken</i>
0	No DST (default)	n/a	n/a	n/a	n/a	none
1	Australian winter	10	25	Sunday	0	Hour advanced 1
33	Australian summer	3	25	Sunday	1	Hour retarded by 1
2	Bahamas winter	4	1	Sunday	0	Hour advanced 1
34	Bahamas summer	10	25	Sunday	1	Hour retarded by 1
3	Brazil winter	11	1	Sunday	0	Hour advanced 1
35	Brazil summer	2	15	Sunday	1	Hour retarded by 1
4	Canada winter	4	1	Sunday	0	Hour advanced 1
36	Canada summer	10	25	Sunday	1	Hour retarded by 1
5	Chile winter	10	8	Saturday	0	Hour advanced 1
37	Chile summer	3	8	Saturday	1	Hour retarded by 1
6	Cuba winter	4	1	Any	0	Hour advanced 1
38	Cuba summer	10	25	Sunday	1	Hour retarded by 1
7	Eastern Europe winter	3	25	Sunday	0	Hour advanced 1
39	Eastern Europe summer	10	25	Sunday	1	Hour retarded by 1
8	Egypt winter	4	24	Friday	0	Hour advanced 1
40	Egypt summer	9	24	Thursday	1	Hour retarded by 1
9	Falklands winter	9	8	Sunday	0	Hour advanced 1
41	Falklands summer	4	6	Sunday	1	Hour retarded by 1
10	Greenland winter	3	25	Sunday	1	Hour advanced 1
42	Greenland summer	10	25	Sunday	2	Hour retarded by 1
11	Iran winter	1	1	Any	0	Hour advanced 1
43	Iran summer	7	1	Any	1	Hour retarded by 1
12	Iraq winter	4	1	Any	0	Hour advanced 1
44	Iraq summer	10	1	Any	1	Hour retarded by 1
13	Israel winter	4	1	Friday	0	Hour advanced 1
45	Israel summer	9	1	Friday	1	Hour retarded by 1
14	Kirgistan winter	3	25	Sunday	0	Hour advanced 1
46	Kirgistan summer	10	25	Sunday	1	Hour retarded by 1
15	Lebanon winter	3	25	Sunday	0	Hour advanced 1
47	Lebanon summer	10	25	Sunday	1	Hour retarded by 1
16	Mexico winter	4	1	Sunday	0	Hour advanced 1
48	Mexico summer	10	25	Sunday	1	Hour retarded by 1
17	Namibia winter	9	1	Sunday	0	Hour advanced 1
49	Namibia summer	4	1	Sunday	1	Hour retarded by 1
18	New Zealand winter	10	1	Sunday	0	Hour advanced 1
50	New Zealand summer	3	15	Sunday	1	Hour retarded by 1
19	Palestine winter	4	15	Friday	0	Hour advanced 1
51	Palestine summer	10	15	Friday	1	Hour retarded by 1
20	Paraguay winter	9	1	Sunday	0	Hour advanced 1
52	Paraguay summer	4	1	Sunday	1	Hour retarded by 1
21	Russia winter	3	25	Sunday	2	Hour advanced 2
53	Russia summer	10	25	Sunday	4	Hour retarded by 2
22	Syria winter	4	1	Any	0	Hour advanced 1
54	Syria summer	10	1	Any	1	Hour retarded by 1
23	Tasmania winter	10	1	Sunday	0	Hour advanced 1
55	Tasmania summer	3	1	Sunday	1	Hour retarded by 1
24	Tonga winter	11	1	Sunday	0	Hour advanced 1
56	Tonga summer	1	25	Sunday	1	Hour retarded by 1
25	USA winter	4	1	Sunday	0	Hour advanced 1
57	USA summer	10	25	Sunday	1	Hour retarded by 1

<i>DSTEvent Value (decimal)</i>	<i>Definition</i>	<i>Month</i>	<i>Date</i>	<i>Day of week</i>	<i>Hour</i>	<i>Action taken</i>
26	Western Europe winter	3	25	Sunday	1	Hour advanced 1
58	Western Europe summer	10	25	Sunday	2	Hour retarded by 1

The 3-byte Bluetooth device class determines what the module claims to be when other Bluetooth devices ask it. It affects the icon that appears on other Bluetooth devices and may affect the device discovery function. In particular some mobile phones only look for certain sub classes, e.g. headsets.

The device class consists of three elements: the services available, the major device class and the minor device class. BlueMatik can be programmed to claim to be capable of any number of services, however exactly one Major Class must be specified. The minor device class is an optional addition, defining a subset of the major device class.

The first two bytes of the device class contain the services information and the major device class. They are calculated by bitwise-ORing together as many services that are required and the one Device Major Class required.

<i>Byte A</i>	<i>Byte B</i>	<i>Description</i>	<i>Data Type</i>
0x00	0x20	Limited discovery mode (default)	Services
0x01	0x00	Positioning	
0x02	0x00	Network (default)	
0x04	0x00	Rendering	
0x08	0x00	Capturing	
0x10	0x00	Object transfer (default)	
0x20	0x00	Audio	
0x40	0x00	Telephony (default)	
0x80	0x00	Information	
0x00	0x01	Computer	Device Major Class
0x00	0x02	Phone (default)	
0x00	0x03	LAN	
0x00	0x04	AV	
0x00	0x05	Peripheral	
0x00	0x06	Imaging	
0x00	0x1F	Uncategorized	
0x00	0x00	Miscellaneous Device Class	

The last byte defines the minor device class. Its interpretation depends on the major device class specified as follows.

Byte C	Computer Major Class	Phone Major Class	LAN Major Class	AV Major Class
0x00	Other	Other	LAN 0% utilized	Other
0x04	Desktop	Cellphone (default)		Wearable headset
0x08	Server	Cordless phone		Hands free device
0x0C	Laptop	Smartphone		
0x10	Handheld	Gateway / modem		Microphone
0x14	Palm-sized	ISDN		Loudspeaker
0x18	Wearable			Headphones
0x1C				Walkman
0x20			LAN 1-17% utilized	Car audio
0x24				Set top box
0x28				Hi-Fi
0x2C				VCR
0x30				Video camera
0x34				Camcorder
0x38				Monitor
0x3C				Monitor with audio
0x40			LAN 17-33% utilized	Conferencing device
0x48				Toy
0x60			LAN 33-50% utilized	
0x80			LAN 50-67% utilized	
0xA0			LAN 67-83% utilized	
0xC0			LAN 83-99% utilized	
0xE0			LAN 100% utilized	

Byte C	Peripheral Device Class <i>Bitwise-OR together one † value and one ‡ value</i>	Imaging Device Class <i>Bitwise-OR together as many values as apply</i>	Uncategorized / Miscellaneous Device Class
0x00	No keyboard or pointing device †		Uncategorized / Miscellaneous
0x00	Other ‡		
0x04	Joystick		
0x08	Gamepad ‡		
0x0C	Remote control ‡		
0x10	Sensing device ‡	Display	
0x14	Digitizer ‡		
0x18	Card reader‡		
0x1C			
0x20		Camera	
0x40	Keyboard but no pointing device †	Scanner	
0x80	Pointing device but no keyboard †	Printer	
0xC0	Keyboard and pointing device †		

The *DateTimeU* structure is defined as follows:

Field name	Datatype	Contains	Range
sec	byte	Second	0 – 59
min	byte	Minute	0 – 59
hour	byte	Hour	0 – 23
date	byte	Date	1 – 31
dow	byte	Day of week	0 – 6 Sunday to Saturday respectively 7 = Unknown

Field name	Datatype	Contains	Range
month	byte	Month	1 – 12
year	uint16	Year	0 – 65535

Examples of Configure DARC-I Commands:

Configure DARC-I Command Examples	
Set authentication security (binary)	0x04 0x01 0x01 0x01
Set authentication security (ASCII)	"04010101<CR><LF>"
Set PIN "1234" (ASCII)	"0801023132333400<CR><LF>"
Set device name "Fred" (ASCII)	"0801034672656400<CR><LF>"
Set binary responses (ASCII)	"04010500<CR><LF>"
Set cellphone device class (ASCII)	"0601089FE204<CR><LF>"
Set time to 13:24:50, April 1st, 2005 (ASCII)	"0B014232180D010004D507<CR><LF>"
Get time (ASCII)	"030143<CR><LF>"
Set stream rate to 51.2µs	0x05 0x01 0x21 0x78 0xFE
Set stream rate to 1/1024th sec	0x05 0x01 0x21 0x1F 0x00
Set stream rate to 1 sec	0x05 0x01 0x21 0xFF 0x7F
Set log rate to daily at midday	0x04 0x01 0x22 0x4C
Add AN0 to the list of the data to be sampled	0x05 0x01 0x23 0x10 0x01
Remove AN0 from the list of the data to be sampled	0x05 0x01 0x23 0x10 0x00
Log to ROM only	0x05 0x01 0x24 0x81 0x81
Log to external memory 0xB0, 0xB2	0x05 0x01 0x24 0x10 0x11

Configure I/O Command

The command byte 0x02 configures the ToothPIC I/O. The byte after the command byte is the *Property Byte*, which specifies the exact I/O property being set. The remaining bytes represent the new property value, as follows:

Configure I/O Command Properties		
Property Byte	Property	Remaining Byte(s)
0x01	A to D channels	Range 00 to 0C = Number of analog to digital channels (from AN0 up).
0x02	Negative voltage reference	00 = Vss is -ve voltage reference (default) 01 = AN2 is -ve voltage reference
0x03	Positive voltage reference	00 = Vdd is +ve voltage reference (default) 01 = AN3 is +ve voltage reference
0x04	PWM time base units	00 = Turn PWM off 01 = PWM on, base time unit is 0.2µs 02 = PWM on, base time unit is 0.8µs 03 = PWM on, base time unit is 3.2µs
0x05	PWM period	Range 00 to FF = PWM period in PWM base time, less one.

Configure I/O Command Properties		
Property Byte	Property	Remaining Byte(s)
0x06	Parallel I/O A function	00 = Not used 02 = 2-bit output (AN11 – AN10) 03 = 3-bit output (AN11 – AN9) 04 = 4-bit output (AN11 – AN8) 05 = 5-bit output (AN11 – AN7) 06 = 6-bit output (AN11 – AN6) 07 = 7-bit output (AN11 – AN5) 12 = 2-bit input (AN11 – AN10) 13 = 3-bit input (AN11 – AN9) 14 = 4-bit input (AN11 – AN8) 15 = 5-bit input (AN11 – AN7) 16 = 6-bit input (AN11 – AN6) 17 = 7-bit input (AN11 – AN5)
0x07	Parallel I/O B function	00 = Not used 02 = 2-bit output (AN4 – AN3) 03 = 3-bit output (AN4 – AN2) 04 = 4-bit output (AN4 – AN1) 05 = 5-bit output (AN4 – AN0) 12 = 2-bit input (AN4 – AN3) 13 = 3-bit input (AN4 – AN2) 14 = 4-bit input (AN4 – AN1) 15 = 5-bit input (AN4 – AN0)
0x08	Parallel I/O C function	00 = Not used 02 = 2-bit output (CCP5, CCP4) 03 = 3-bit output (CCP5, CCP4, RxD) 04 = 4-bit output (CCP5, CCP4, RxD, TxD) 05 = 5-bit output (CCP5, CCP4, RxD, TxD, CCP3) 12 = 2-bit input (CCP5, CCP4) 13 = 3-bit input (CCP5, CCP4, RxD) 14 = 4-bit input (CCP5, CCP4, RxD, TxD) 15 = 5-bit input (CCP5, CCP4, RxD, TxD, CCP3)
0x10 – 0x1B	AN0 – AN11 function (0x10 = AN0, 0x11 = AN1, etc)	00 = Digital input (default) 01 = Digital output (Ignored if configured for A to D input.)
0x1C – 0x20	CCP1 – CCP5 function	00 = Digital input (default) 01 = Digital output 02 = PWM output
0x21 – 0x22	INT0 – INT1 function	00 = Digital input (default) 01 = Digital output
0x23	SCL function	As INT0
0x24	SDA function	As INT0
0x25	SDO function	As INT0
0x26	TxD function	As INT0
0x27	RxD function	As INT0

Notes:

I/O pin functions: Pin specifications are ignored for AN inputs if the *A to D channels* property dictates that they should be A to D inputs. I/O pin functions must not be sent for pins used for parallel I/O, external memory and flow control purposes.

Parallel I/O: Pins are modified or read at exactly the same instant.

PWM base time units: Base time units for PWM values. PWM period is in PWM base time units (1 to 256). PWM duty cycle is in quarter PWM base time units (0 to 1023). *PWM time base units* turns PWM outputs on or off, so during initialization, set up period and initial output values first.

Configure I/O Command Examples	
Set AN0 – AN4 as A to D pins (binary)	0x04 0x02 0x01 0x05
Set AN0 – AN4 as A to D pins (ASCII)	"04020105<CR><LF>"
Set: PWM time base 3.2µs PWM period as 256 (=1220Hz) CCP2 pin as PWM (binary)	0x04 0x02 0x04 0x03 0x04 0x02 0x05 0xFF 0x04 0x02 0x1D 0x02

Set I/O Command

The command byte 0x03 sets a ToothPIC I/O output value. The byte after the command byte is the *Property Byte*, which specifies the exact I/O value being set. The remaining bytes represent the new I/O value, as follows:

Set I/O Command Properties		
Property Byte	Property	Remaining Byte(s)
0x06	Parallel I/O A output	Range 00 to 7F = new value
0x07	Parallel I/O B output	Range 00 to 1F = new value
0x08	Parallel I/O C output	Range 00 to 1F = new value
0x10 – 0x1B	AN0 – AN11 output	00 = low 01 = high
0x1C – 0x20	CCP1 – CCP5 output	If digital, as AN0 – AN11. If PWM, range 0000 to 03FF. (Both bytes must be specified.)
0x21 – 0x22	INT0 – INT1 output	As AN0
0x23	SCL output	As AN0
0x24	SDA output	As AN0
0x25	SDO output	As AN0
0x26	TxD output	As AN0
0x27	RxD output	As AN0
0x30	Green LED	00 = off 01 = on
0x31	Red LED	00 = off 01 = on

If the pin is not already configured as an output, the value will be ignored. I/O pin value commands must not be sent for pins used for external memory and flow control purposes.

Set I/O Command Examples	
Set AN10 as high (binary)	0x04 0x03 0x1A 0x01
Set AN10 as high (ASCII)	"04031A01<CR><LF>"
Set CCP2 as 33% duty cycle (binary)	0x04 0x03 0x1D 0x01 0x55

Get I/O Command

The command byte 0x04 requests a ToothPIC I/O value. The byte after the command byte is the *Property Byte*, which specifies the exact I/O value being requested. The value will read and transmitted as an I/O value response.

Get I/O Command Properties	
Property Byte	Property
0x06	Parallel I/O A input
0x07	Parallel I/O B input
0x08	Parallel I/O C input
0x10 – 0x1B	AN0 – AN11 input
0x1C – 0x20	CCP1 – CCP5 input
0x21 – 0x22	INT0 – INT1 input
0x23	SCL input
0x24	SDA input
0x25	SDO input
0x26	TxD input
0x27	RxD input
0x30	Green LED
0x31	Red LED
0x32	Pushbutton

If the pin is configured as digital output, the result will be the current output value. If the pin is configured as CCP output, the result will be unpredictable. I/O pin value commands must not be sent for pins used for external memory and flow control purposes.

Get I/O Command Examples	
Get AN10 (binary)	0x03 0x04 0x1A
Get AN10 (ASCII)	"03041A<CR><LF>"
Get CCP2 (binary)	0x03 0x04 0x1D

Get Memory Command

The Get Memory Data Command byte 0x05 retrieves data from internal or external memory. Very little error checking is done with this command, so it should be used with care.

The byte after the command byte is the `mStR` memory storage type as defined in the *Set Memory Command*. The next two bytes are the `Addr` memory address (little-endian, i.e. least significant byte first) as defined in the *Set Memory Command*.

The final byte is the number of bytes required, up to 120 bytes in ASCII response mode and 248 bytes in binary response mode.

Each Get Memory Command will receive exactly one Got Memory Response.

Get Control Data Command Example	
Get 4 bytes of EE memory from 0x0123	0x06 0x05 0x03 0x23 0x01 0x04

Set Memory Command

The Set Memory Data Command byte 0x06 sets data in internal or external memory. Very little error checking is done with this command, so it should be used with care. It is your responsibility to ensure that you do not overwrite important memory regions. Overwriting `ROM00` locations below 0xC000 may prevent Wireless Field Programming from functioning.

The byte after the command byte is the `mStR` memory storage type as defined below. The next two bytes are the `Addr` memory address (little-endian, i.e. least significant byte first) as defined below.

The remaining bytes are the data to be written to memory `mStr` starting at location `Addr`. Up to 120 bytes may be set with an ASCII Set Memory command and 248 bytes may be set with a binary Set Memory command.

Set Memory Command Example	
Set EE memory from 0x0123 with the values 0x05 0x06 0x07 0x08.	0x09 0x06 0x03 0x23 0x01 0x05 0x06 0x07 0x08

The following regions of memory are not used and you may use them as you wish:

Type	mStr value	Addr min	Addr max
RAM	0x02	0x0000	0x08FF
EE	0x03	0x0000	0x3FF
External memory I2C address 0xB0 / 0xB1	0x10	0x0000	IC limit
External memory I2C address 0xB2 / 0xB3	0x11	0x0000	IC limit
External memory I2C address 0xB4 / 0xB5	0x12	0x0000	IC limit
External memory I2C address 0xB6 / 0xB7	0x13	0x0000	IC limit
External memory I2C address 0xB8 / 0xB9	0x14	0x0000	IC limit
External memory I2C address 0xBA / 0xBB	0x15	0x0000	IC limit
External memory I2C address 0xBC / 0xBD	0x16	0x0000	IC limit
External memory I2C address 0xBE / 0xBF	0x17	0x0000	IC limit
ROM (Flash)	0x81	0x0000	0xEFFF

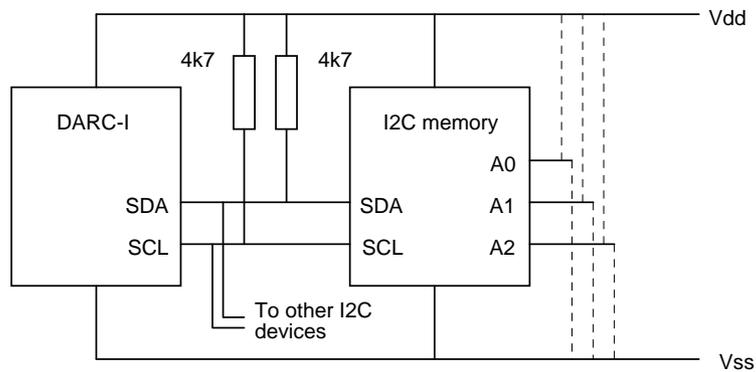
Notes:

RAM memory may be written to an unlimited number of times. Its contents are lost when power is turned off.

ROM memory may be written to approximately 100,000 times. Its contents are retained when power is turned off. Writing to ROM causes the main microcontroller clock (but not the real time clock) to be suspended for about 2ms.

EE memory may be written to approximately 1,000,000 times. Its contents are retained when power is turned off.

External memory must be installed in order to use it. External memory access has been tested with Microchip Technology 24Cxxx series EEPROM devices and external I2C memory devices should use the same I2C communications format. Setting up external I2C memory is as follows, with reference to the figure:



1. Connect the Vdd, Vss , SDA and SCL lines for all memory devices.
2. Provide 4K7 pullup resistors for SDA and SCL.
3. Hardwire each I2C memory device address line A0-A2 to Vcc or Vss to specify separate storage addresses as in the following table. (24C512-compatible devices should be connected to $0x1y$ where y is even; the upper half of the memory is accessed as $0x1y+1$.)

Storage address	A2	A1	A0
0x10	Vss	Vss	Vss
0x11	Vss	Vss	Vdd
0x12	Vss	Vdd	Vss
0x13	Vss	Vdd	Vdd
0x14	Vdd	Vss	Vss
0x15	Vdd	Vss	Vdd
0x16	Vdd	Vdd	Vss
0x17	Vdd	Vdd	Vdd

4. Send an I2C memory setup to initialize the I2C memory management.
5. Call Get Memory and Set Memory as required.

Stream Data Command

The Stream Data Command byte $0x07$ samples inputs and sends the results immediately to the remote device. There are no bytes after the command byte. The next two bytes are the `Addr` memory address (little-endian, i.e. least significant byte first) as defined below.

The sampling rate, between $1/32768$ second and 2 seconds, is defined by the Stream / Frame Rate property set using the Configure DARC-I command.

The Stream / Frame / Log Inputs property, set using the Configure DARC-I command, specifies what is recorded. This also defines the size of each record and the maximum speed at which data can be recorded. Each record will contain a byte for each digital/parallel value recorded and two bytes (lowest byte first) for each analog value recorded. The order will be as follows (only those inputs specified by the Stream / Frame / Log Inputs property will be recorded):

Input ID value	Property
0x06	Parallel I/O A input
0x07	Parallel I/O B input
0x08	Parallel I/O C input
0x10 - 0x1B	AN0 - AN11 input
0x1C - 0x20	CCP1 - CCP5 input
0x21 - 0x22	INT0 - INT1 input

Input ID value	Property
0x23	SCL input
0x24	SDA input
0x25	SDO input
0x26	TxD input
0x27	RxD input

Digital and parallel inputs are sampled more or less instantly. Analog values require about 20µs each. Values are sampled sequentially, not simultaneously.

The sample records are transmitted immediately to the remote device. Samples will continue to be transmitted until any character is transmitted from the remote device. That character will then be discarded and should not be the beginning of a command. The DARC-I will then respond with an '0201' OK response.

If, during streaming, the data cannot be transmitted fast enough, no further data will be transmitted. The DARC-I will then wait for any character is transmitted from the remote device and reply with a '0302F4' overflow error.

Data streaming cannot take place while data logging is in operation and if a Frame Data Command is sent when *LogRate* is non-zero, an '0302F1' error will be generated.

Notes:

- Streaming is not possible while the Real Time Clock is operating. The Real Time Clock is turned off when the Stream Data command is received and it is not turned off again.
- For reasons of speed, the output stream is not buffered and it is assumed that the remote device can absorb the data as fast as it is transmitted.

Stream Data Example	
Set AN0 and AN1 as analog	0x04 0x02 0x01 0x02
Record AN0	0x05 0x01 0x23 0x10 0x01
Record AN1	0x05 0x01 0x23 0x11 0x01
Set 1s sample rate	0x05 0x01 0x21 0x88 0x13
Start streaming	0x02 0x07
<i>Stream data response</i>	<i>0x42 0x01 0x36 0x02</i>
↓	↓
<i>Stream data response</i>	<i>0xA2 0x00 0x25 0x02</i>
Stop byte	0x01
<i>OK response</i>	<i>0x02 0x01</i>

Frame Data Command

The Frame Data Command byte 0x08 samples a fixed number of times. The sampling rate, between 1/32768 second and 2 seconds, is defined by the Stream / Frame Rate property set using the Configure DARC-I command. The two bytes after the command byte are the number of samples to take.

Each sample record structure is as defined by the Stream / Frame / Log Inputs property (see Stream Data Command). The data are stored as defined for the Set Frame / Log Memory Locations property of the Configure DARC-I command. Each Frame Data command will restart storing data at the beginning of *FLMemStart*.

When the command is complete, the DARC-I will then respond with an '0201' OK response, or an '0302F4' overflow error if the data could not be sampled fast enough. Data framing cannot take place while data logging is in operation and if a Frame Data Command is sent when *LogRate* is non-zero, an '0302F1' error will be generated.

Note: Framing in not possible while the Real Time Clock is operating. The Real Time Clock is turned off when the Frame Data command is received and it is not turned off again..

Frame Data Example	
Set AN0 and AN1 as analog	0x04 0x02 0x01 0x02
OK response	0x02 0x01
Record AN0	0x05 0x01 0x23 0x10 0x01
OK response	0x02 0x01
Record AN1	0x05 0x01 0x23 0x11 0x01
OK response	0x02 0x01
Set 1ms sample rate	0x05 0x01 0x21 0x05 0x00
OK response	0x02 0x01
Log to RAM	0x05 0x01 0x24 0x02 0x02
OK response	0x02 0x01
Record 512 values	0x04 0x08 0x00 0x02
OK response	0x02 0x01
<i>(Data now in RAM locations 0x000 to 0x8FF)</i>	

Logging Data

There is no Log Data command. Data logging begins as soon as the **LogRate** configuration property is non-zero and continues while it remains non-zero.

The sampling rate is defined by the **LogRate** value. The data are stored as defined for the Set Frame / Log Memory Locations property of the Configure DARC-I command. Each **LogRate** is changed, logging will restart at the beginning of **FLMemStart**.

Each record will consist of the 8-byte `DateTimeU` clock time (defined for the Set Date Time property) followed by a sample record structure is as defined by the Stream / Frame / Log Inputs property (see Stream Data Command).

Log Data Example	
Set AN0 and AN1 as analog	0x04 0x02 0x01 0x02
OK response	0x02 0x01
Record AN0	0x05 0x01 0x23 0x10 0x01
OK response	0x02 0x01
Record AN1	0x05 0x01 0x23 0x11 0x01
OK response	0x02 0x01
Log to RAM	0x05 0x01 0x24 0x02 0x02
OK response	0x02 0x01
Set log to once per sec	0x04 0x01 0x22 0x01
OK response	0x02 0x01
<i>(logs to RAM 0x000)</i>	<i>32 18 0D 01 05 04 D5 07 36 01 3B 03</i>
<i>(logs to RAM 0x00C)</i>	<i>33 18 0D 01 05 04 D5 07 74 01 19 03</i>
<i>(logs to RAM 0x018)</i>	<i>34 18 0D 01 05 04 D5 07 3C 01 44 03</i>
<i>(logs to RAM 0x024)</i>	<i>35 18 0D 01 05 04 D5 07 49 01 C0 03</i>
Stop logging	0x04 0x01 0x22 0x00
OK response	0x02 0x01
<i>(Data now in RAM locations 0x000 to 0x030)</i>	

Responses

Responses length will vary according to the length of the command sent. The first byte is the *response length byte*, equal to the total number of bytes in the message. The second byte is the *response byte*, which indicates how the remainder of the message should be interpreted.

Responses may be in either ASCII or binary as specified by the *ASCII responses* property. In ASCII format, each byte is transmitted as a two hexadecimal digits and the entire command will be preceded and followed by a <CR><LF> pair (i.e. the control characters 0x0D and 0x0A).

If the *Responses anytime* property is set, all commands generate a response. This will be the OK response if no other response is appropriate.

Command Summary		
Response	Response Byte	Interpretation
OK	0x01	Acknowledges completion of previous command.
Error	0x02	Reports an error
Date Time	0x03	Reports the time and date
I/O Value	0x04	The value of the requested I/O pin
Got Memory	0x05	The value of the requested memory location.
Initialization	0x44	A message sent to indicate initialization is complete.

OK Response

The response byte 0x01 indicates that the previous command has been processed and another command may be sent.

OK Response Examples	
OK (binary)	0x02 0x01
OK (ASCII)	" <CR><LF>0201<CR><LF>"

Error Response

The response byte 0x02 indicates that an error has occurred. The byte after the response byte is the *Error Byte*, which specifies exact error that occurred:

Error Response Error Values		
Error Byte	Error Name	Interpretation
0xF1	Not understood	The previous command was not understood.
0xF4	Overflow	Data could not be sampled as fast as requested.

In addition, if the *On internal error...* property specifies that internal errors should be reported to the host, the error byte may equal any of the *ErrorStatus*, *BMTE_Error*, *BMTE_Warning* error values as defined for the ToothPIC product; however, no such errors are expected.

Error Response Examples	
Not understood (ASCII)	" <CR><LF>0302F1<CR><LF>"
ErrorStatus error Memory Failure (binary)	0x03 0x02 0x0B

Date Time Response

The response byte 0x03 reports the current date and time to the host. . The 8 bytes after the response byte are the date and time in `DateTimeU` format as defined for the Set Date Time property.

Date Time Response Examples – 13:24:50, Friday, April 1st, 2005	
Date Time (binary)	0x0A 0x03 0x32 0x18 0x0D 0x01 0x05 0x04 0xD5 0x07
Date Time (ASCII)	"<CR><LF>0A0332180D010504D507<CR><LF>"

I/O Value Response

The response byte 0x04 reports an I/O input value to the host. The first byte after the response byte indicates which value is being reported and matches those used in the Get I/O Command. The remaining byte(s) will contain the reported value. Its size will vary according to data type as shown in the examples below.

I/O Value Response Examples	
SDO digital input high (binary)	0x04 0x04 0x25 0x01
SDO digital input low (ASCII)	"<CR><LF>04042500<CR><LF>"
AN3 analog input 0x234 (range 0 to 1023) (ASCII)	"<CR><LF>0504130234<CR><LF>"
Parallel A (7-bit) analog input 0x14 (range 0 to 31) (ASCII)	"<CR><LF>04040614<CR><LF>"

Got Memory Response

Control Data Response byte 0x05 is a response to a Get Memory Data command.

The byte after the command byte is the `mStr` memory storage type as defined for the Set Memory command. The next two bytes are the `Addr` memory address (little-endian) as defined for the Set Memory command.

The remaining bytes are the data read from memory `mStr` starting at location `Addr`, as requested

Got Memory Response Example	
Report EE memory from 0x0123 being values 0x05 0x06 0x07 0x08.	0x09 0x09 0x03 0x23 0x01 0x05 0x06 0x07 0x08

Initialization Response

The initialization response byte 0x44 indicates that the DARC-I has initialized correctly. The 15 bytes of the response including the response length byte will comprise the bytes 0x11, 0x44, 0x41, 0x52, 0x43, 0x2D, 0x49, 0x20 followed by the ToothPIC Services version number as ASCII characters.

Initialization Response Examples – Version 3.0.0004	
Initialization complete (binary)	0x10 then "DARC-I 3.0.0004"
Initialization complete (ASCII)	"<CR><LF>11444152432D4920 332E302E3030303034<CR><LF>"

Command / Response User Guide

The following tables of groups of commands and responses show how to achieve common tasks with DARC-I.

Initialization	
Response	Meaning
11444152432D4920332E302E3030303034	Initialization successful, version 3.0.00004

Setting SDO as digital output		
Command	Response	Meaning
04022501	0201	Set SDO to output
04032501	0201	Set output value to high

Setting SDO as digital input		
Command	Response	Meaning
04022300	0201	Set SCL to input
030423	04042300	Get input value, result 0x00 = low

Setting CCP1 as CCP output		
Command	Response	Meaning
04020403	0201	Set PWM time base to 3.2 μ s
040205FF	0201	Set PWM period to $(0xFF+1) \times 3.2\mu s = 819.2\mu s$ (1220Hz)
04021C02	0201	Set CCP1 to PWM output
05031C01FF	0201	Set duty cycle to $0x01FF \times (3.2\mu s / 4) = 408.8\mu s$

Setting AN1 as analog input		
Command	Response	Meaning
04020102	0201	Set AN0 and AN1 as analog inputs
030411	05041103EB	Read AN1,

Setting AN11, AN10, AN9 as parallel output		
Command	Response	Meaning
04020603	0201	Set parallel output A to 3-bit
04030605	0201	Set output to $0x05 = 101$ binary

Setting AN11, AN10, AN9 as parallel input		
Command	Response	Meaning
04020613	0201	Set parallel input A to 3-bit
030406	04040603	Read input value, result $0x03 = 011$ binary

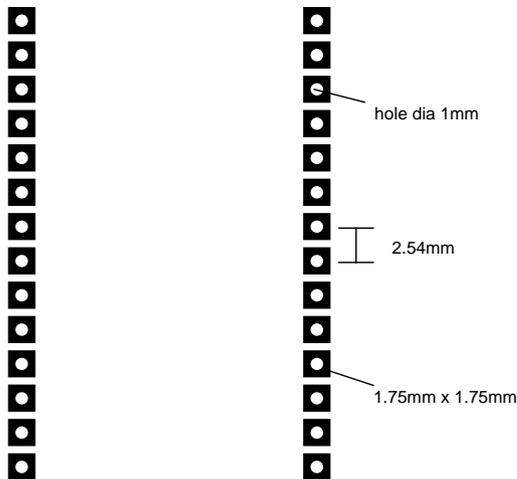
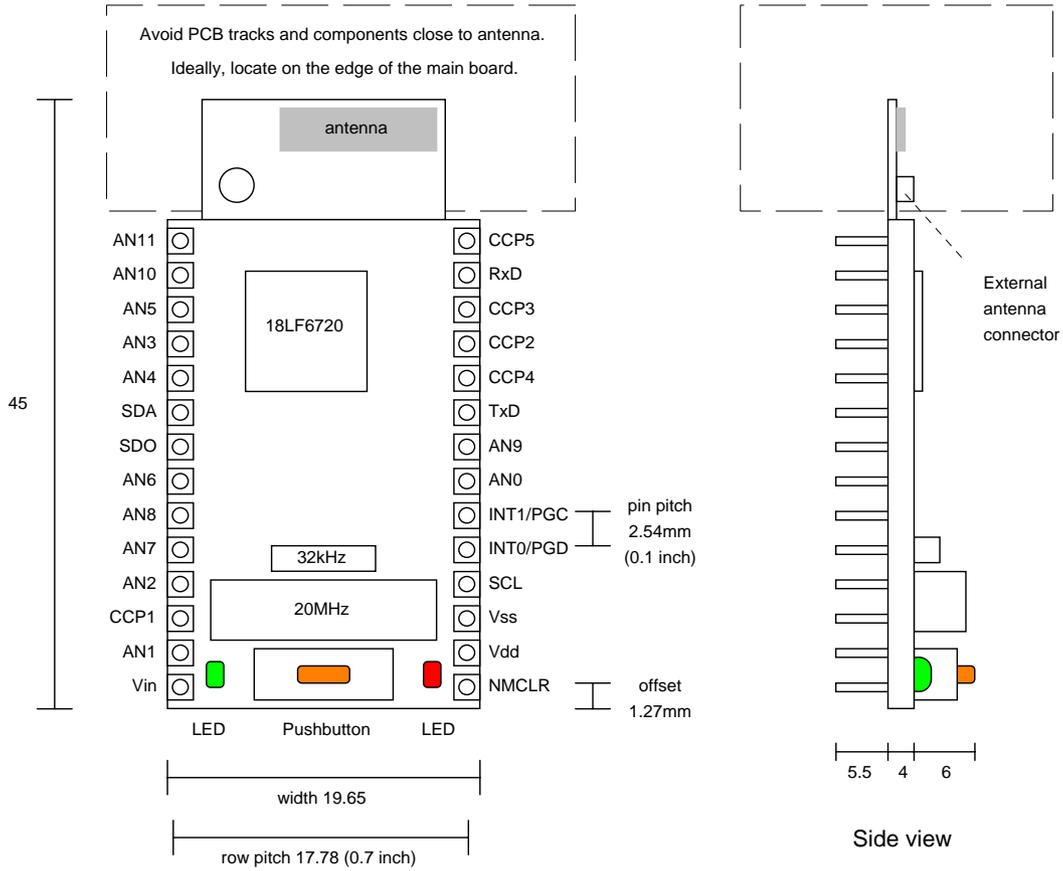
Customization

The DARC-I module is capable of setting up and controlling I/O and logging data in various ways. In some commercial cases you may wish to customize the DARC module. For example, add power saving features or increase the maximum speed of data acquisition.

To make customization possible, we have made the C source code freely available for you to customize. To do this you will need to know how to program microcontrollers in C and also be familiar with Microchip Technology's MPLAB development environment. Information for customizing the DARC-I is given in the ToothPIC documentation available from www.FlexiPanel.com.

Since many customization requirements are relatively simple, we offer a paid-for customization service to commercial developers who are not familiar with programming in the MPLAB development environment. Please contact us for details.

Mechanical Data



Main board PCB pad layout

Dimensions in mm unless otherwise stated

Notes: Ensure the area where the module is mounted has a solid ground plane. To remove the module from an IC socket or breadboard, lever it out using a screwdriver against the pin headers at the sides. Levering from either end may damage components.

Technical Specifications

Max operating temperature	-20°C to +75 °C
Max storage temperature	-30°C to +85 °C
Dimensions L x W x H	45mm x 20mm x 10mm excluding pins

Electrical

Supply Voltage (unregulated)	5V to 10V
Supply Voltage (regulated)	4.5V to 5.5V
Peak power requirement excluding draw on I/O pins	270mA
Maximum current on any I/O pin	25mA
Maximum total current on all I/O pins	200mA
Max voltage on I/O pins	-0.5V to +5.5V

Radio

Max RF output power	Class I = 100mW = +20dBm
RF frequency range	2402MHz to 2480MHz
RF channels / frequency hop rate	79 / 1600 Hz
Range	100m nominal
Communication latency, μ P to μ P	30ms to 50ms
Maximum data rate	50-90 Kbaud depending on conditions
Pairing method	Unit key pairing

FCC, CE and IC modular approval

The radio has 'modular approval' for USA, Canada and certain European countries, provided the existing integral antenna is used. The CE mark on the module indicates that it does not require further R&TTE certification. The exterior of the product should be marked as follows:

Contains Transmitter Module FCC ID: CWTUGPZ1 Contains Transmitter Module IC: 1788F-UGPZ1

Ordering Contact Details

DARC-I is manufactured and distributed by



R F Solutions Ltd
Unit 21, Cliffe Industrial Estate,
Lewes, E. Sussex BN8 6JL, United Kingdom
email : sales@rfsolutions.co.uk
<http://www.rfsolutions.co.uk>
Tel: +44 (0)1273 898 000, Fax: +44 (0)1273 480 661

Technical Support and Customization

DARC-I is owned and designed by FlexiPanel Ltd:



FlexiPanel Ltd
Suite 120, Westbourne Studios
242 Acklam Road
London W10 5JJ, United Kingdom
Tel +44 (0) 20 7524 7774
email: support@flexipanel.com