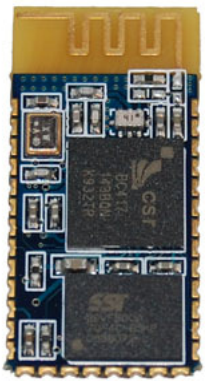
Emartee

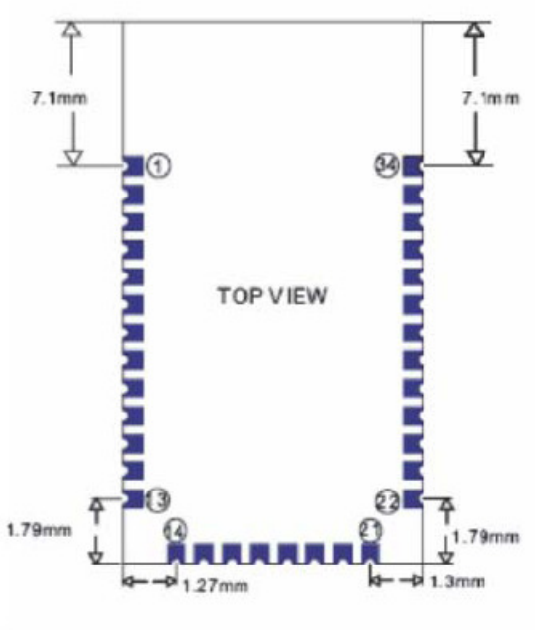
CuteDigi BMX Bluetooth to UART/I2C/USB Module (GEN II)

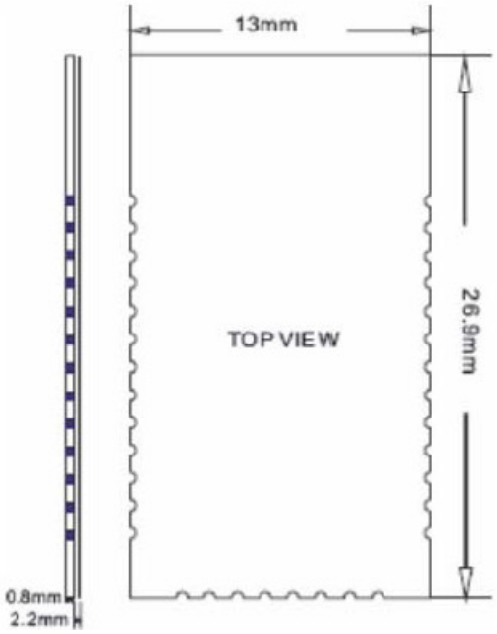


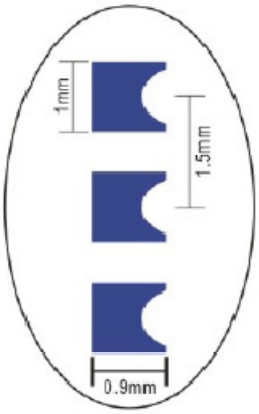
CuteDigi BMX Bluetooth to UART module uses CSR BlueCore4- External chipsets. It embeds 8Mbit flash for software storage, and supports 3.3V power supply. BMX is a muti-function module. It can be used in different products according to the embedded firmware settings. It is especially targeted for data transfer.

The second generation Bluetooth UART module has two working mode: AT command mode, and automatic binding transparent data mode. In automatic binding data transparent mode, it can be configured to Master, Slave or Loopback three different modes, and it will connect to or be connected by other devices that support SPP protocol per configuration. In AT command mode, user can configure the module and send control commands. By controlling logical level of IO pin PIO11, user can switch the working modes between AT command mode and transparent data mode.

※ Chipset：CSR BC417143 (BlueCore4- External)
※ Bluetooth version：V2.0+EDR
※ Output power：Class II
※ Flash：8Mbit
※ Power Supply：3.3V
※ Interface：I2C，UART，PCM，USB1.2
※ Size：26.9mm*13mm*2.2mm
※ Rohs: Yes







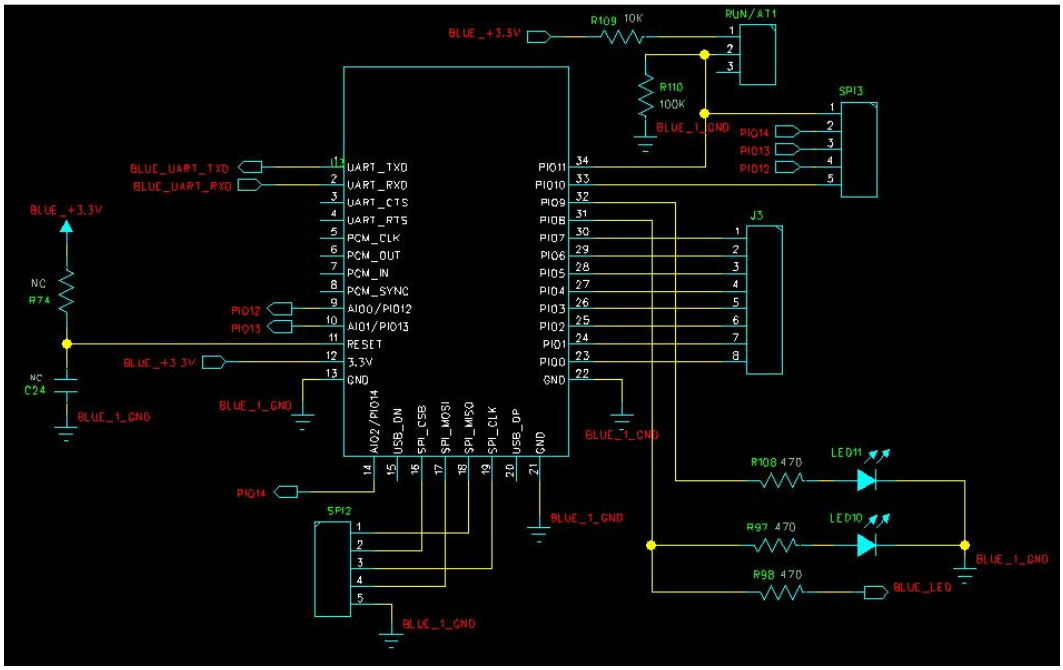
NO	PIN NAME	NO	PIN NAME	NO	PIN NAME
1	UART-TX	13	GND	25	PIO(2)
2	UART-RX	14	GND	26	PIO(3)
3	UART-CTS	15	USB D-	27	PIO(4)
4	UART-RTS	16	SPI-CSB	28	PIO(5)
5	PCM-CLK	17	SPI-MOSI	29	PIO(6)
6	PCM-OUT	18	SPI-MISO	30	PIO(7)
7	PCM-IN	19	SPI-CLK	31	PIO(8)
8	PCM-SYNC	20	USB D+	32	PIO(9)
9	AIO(0)	21	GND	33	PIO(10)
10	AIO(1)	22	GND	34	PIO(11)
11	RESET	23	PIO(0)		
12	3.3V	24	PIO(1)		

Other pins used by Bluetooth UART module:

1. PIO8 is used to control LED indicating the status. It will blink after power on. Different blink intervals are used to indicate different status.
2. PIO9 is used to control LED indicating paring. It will be steady on when paring is successful.
3. PIO11 is used to switch the working mode. High level-> AT command mode; Floating or low level-> normal transparent data mode.
4. The module has built-in power on reset circuitry.

Steps to configure the module as master device:

1. PIO11 is set to high.
2. Power on, and the module will enter into AT command mode.
3. Use hyper terminal or other serial console, set baud rate 38400, data 8 bit, stop bit 1, no parity, no flow control.
4. Send “AT+ROLE=1\r\n”, if successful, it will return “OK\r\n”.
5. PIO is set to low, and power cycle, the module will be master device and automatically search for slave device and do binding.



UART communication circuit

AT Commands

(AT commands can be upper or lower case, and also end with \r\n)

#1 : Test Comamnd

Command	Return	Argument
AT	OK	NONE

#2 : Reset

Command	Return	Argument
AT+RESET	OK	NONE

Results: It works as power cycle.

#3: Poll the software version

Command	Return	Argument
AT+VERSION?	+VERSION:<Param OK	Param: software version

Example:
at+version?\r\n
+VERSION:1.0-20090818
OK

#4: Restore the default setting

Command	Return	Argument
AT+ORGL	OK	NONE

Restore the default setting:

1. Device class: 0
2. Inquiry code: 0x009e8b33
3. Device mode: Slave mode
4. Binding mode: SPP
5. Serial port: 38400 bits/s; 1 stop bit, no parity
6. Pairing code: “1234”
7. Device name: “HHW-SPP-1800-2

#5: Poll the address of the Bluetooth device

Command	Return	Argument
AT+ADDR?	+ADDR: <Param> OK	Param: the address of the Bluetooth device

Representation of the address: NAP:UAP:LAP (HEX)
Examples:
The address of the Bluetooth device is: 12:34:56:ab:cd:ef

At+addr?\r\n
+ADDR:1234:56:abcdef
OK

#6: Set and poll device name

Command	Return	Argument
AT+NAME=<Para1>	OK	Param: device name Default: "HHW-SPP-1800-2"
AT+NAME?	1: +NAME: <Param> OK --- successful 2: FAIL --- fail	

Example:

AT+NAME=HHW-SPP-1800-2\r\n ———————— Set Device name as HHW-SPP-1800-2

OK

AT + NAME="HHW-SPP-1800-2"\r\n ———————— Set Device name as HHW-SPP-1800-2

OK

at+name?\r\n

+NAME: Beijin

OK

#7: Poll remote device name

Command	Return	Argument
AT+RNAME? <Param1>	1: +RNAME: <Param2 > OK --- successful 2: FAIL --- fail	Param1: remote device address Param2: remote device name

Representation of the address: NAP:UAP:LAP (HEX)
Examples:
The address of the remote Bluetooth device is: 00:02:72:od:22:24, the device name is: Bluetooth

t+rname? 0002,72,0d2224\r\n
+RNAMELBluetooth
OK

#8: Set/Poll device role

Command	Return	Argument
AT+ROLE= <Param>	OK	Param: 0 – slave 1 – Master 2 – Slave-loop Default: 0
AT+ROLE?	+ROLE: <Param > OK	

Explanation of device roles:
Slave – be connected by other device
Slave-loop – be connected by other device, receive and send back whatever received
Master – Actively poll the nearby device and initialize binding to other devices.

#9: Set and poll device type

Command	Return	Argument
AT+CLASS=<Param>	OK	Param: device type Device type is a 32-bit parameter. It is used to indicate the device class and the service it supports Default: 0 The actual meaning is explained in appendix 1.
AT+CLASS?	1. +CLASS: <Param> OK 2. FAIL	

In order the effectively filter the nearby device and quickly locate the users self defined device, user can set the device to be nonstandard device, such as 0x1f1f (hex)

#10: Set/Poll Inquire Access Code

Command	Return	Argument
AT+IAC=<Param>	1: OK 2: FAIL	Param: Inquire Access Code Default: 938b33 Detailed explanation can be found the appendix.
AT+IAC?	+IAC: <Param> OK	

If the inquire access code is set to GIAC(General Inquire Access Code: 0x9e8b33), it can be used to discover or be discovered by all nearby devices. If user wants the device to be able to be found quickly, user can set the Inquire Access Code to be code not as GIAC and LIAC, such as 0x928b3f.

Example:
AT+IAC=928b3f\r\n
OK
AT+IAC?\r\n
+ IAC: 928b3f
OK

#11: Set and poll Inquiry mode

Command	Return	Argument
AT+INQM=<Param1>, <Param2>, <Param3>	1. OK 2. FAIL	Param1: Inquiry Mode 0— inquirey mode standard 1— inquiry mode rssi Param2: max response number Param3: time out, 1-48 (1.28s-61.44s) Default: 1,1,48
AT+INQM?	+INQM: <Param1>, <Param2>,<Param3> OK	

AT+INQM=1,9,48\r\n -- Set inquiry mode: with RSSI, max device response number 9 then stop inquiry, max time out 48X1.28=61.44s
OK
AT+INQM?\r\n
+INQM:1,9,48
OK

#12: Set and poll paring password

Command	Return	Argument
AT+PSWD=<Param>	OK	Param: paring password Default: “1234”
AT+PSWD?	+PSWD:<Param> OK	

#14: Set and poll serial port parameters

Command	Return	Argument
AT+UART=<Param1>,<Param2>,<Param3>	OK	Param1: baud rate (bits/s) 4800 9600 19200 38400 57600 115200 230400 460800 912600 1382400 Param2: stop bit 0- 1 bit 1- 2 bits Param3: parity bit 0- None 1- Odd 2- Even Default: 9600,0,0
AT+UART?	+UART:<Param1>,<Param2>,<Param3> OK	

Example: Set serial port parameters to 115200, 2 bits stop bit, and even parity

AT+UART=115200, 1,2 \r\n

OK

AT+UART?

+UART:115200,1,2

OK

#14: Set and poll connection mode

Command	Return	Argument
AT+CMODE=<Param>	OK	Param: 0 – specific address mode (the address is specified in binding command) 2- No specific address Default: 0
AT+CMODE?	+CMODE::<Param> OK	

#15: Set and poll binding device address

Command	Return	Argument
AT+BIND=<Para1>	OK	Param – Binding Bluetooth device address Default address: 00:00:00:00:00:00
AT+BIND?	+BIND:<Param> OK	

The address can be represented as NAP:UAP:LAP (hex)
The binding command is only valid in specific address mode.

Example:
AT+BIND=1234,56,abcdef\r\n
OK
AT+BIND?\r\n
+BIND:1234:56:abcdef
OK

#16: Set/Poll the polarity of LED indicator driver

Command	Return	Argument
AT+POLAR=<Param1>, <Param2>	OK	Param1: 0 – PI08 outputs low level to turn on LED 1- PI08 outputs high level to turn on LED Param2: 0-PI09 outputs low level to turn on LED 1-PI09 outputs high level to turn on LED Default: 1,1
AT+DEFAULT		

PI08 drives the working status, and PI09 drives the link status.

Example:
PI08 outputs low level to turn on LED, and PI09 outputs high level to turn on LED.
AT+POLAR=0,1 \r\n
OK
AT+POLAR?\r\n
+POLAR:0,1
OK

#17: Set single PIO output

Command	Return	Argument
AT+PIO=<Param1>,<Param2>	OK	Param1: PIO port number (decimal) Param2L PIO port output 0- Low voltage 1- High voltage

The useable port is PIO2- PIO7 and PIO10.

Example:
1. PIO10 outputs high level
   AT+PIO=10,1\r\n
   OK
2. PIO10 outputs low level
   AT+PIO=10,0\r\n
   OK

#18: Set multiple port output

Command	Return	Argument
AT+MPIO=<Param>	OK	Param: PIO port number mask combination (hex)

The useable port is PIO2- PIO7 and PIO10.
PIO port mask = (1 << port number)
PIO port mask combination = (PIO port mask 1| PIO port mask 2 |PIO port mask 3 |...)

Example:
PIO2 mask= (1<<2)=0x004
PIO10 mask = (1<<10)=0x400
PIO port mask combination= (0x004 | 0x400)=0x404
PIO 2 and PIO 10 output high:
AT+MPIO=404\r\n
OK

#19: Poll PIO port input

Command	Return	Argument
AT+MPIO?	+MPIO: <Param> OK	Param- PIO port value (16 bits) Param[0]=PIO0 Param[1]=PIO1 Paramp2]=PIO2 … Param[10]=PIO10 Param[11]=PIO11

#20:Set/Poll Inquiry parameters

Command	Return	Argument
AT+IPSCAN=<Param1>,<Param2>, <Param3>, <Param4>	OK	Param1: inquiry time interval Param2:continous poll time Param3: call time interval Param4: call continuous time All above are decimal numbers Default: 1024, 512, 1024, 512
AT+IPSCAN?	+IPSCAN:<Param1>, <Param2>,<Param3>,<Param4>	

#21:Set/Poll SNIFF energy saving parameters

Command	Return	Argument
AT+SNIFF=<Param1>,<Param2>,<Param3>,<Param4>	OK	Param1: max time Param2: min time Param3: try time Param4: time out All above are decimal numbers Default: 0,0,0,0
AT+SNIFF?	+SNIFF:<Param1>,<Param2>,<Param3>,<Param4>	

#22: Set/Poll Security and Encryption modes

Command	Return	Argument
AT+SENM=<Param1>,<Param2>	1: OK 2:FAIL	Param1: Security mode 0- Sec_mode0_off 1- Sec_mode1_non-secure 2- Sec_mode2_service 3- Sec_mode3_link 4- Sec_mod_unknown Param2:encryption mode 0- hci_enc_mode_off 1- hci_enc_mode_pt_to_pt 2- hci_enc_mode_pt_to_pt_and_bcast Default: 0,0
AT+SENM?	+SENM:<Param1>,<Param2> OK	

#23: Delete Authenticated Device from the authenticated device list

Command	Return	Argument
AT+RMSAD=<Param>	OK	Param: Bluetooth device address

Example:
Delete device with address: 12:34:56:ab:cd:ef
at+rmsad=1234:56:abcdef\r\n
OK

Or
at+rmsad=1234:56:abcdef\r\n
FAIL ==== there is no such device in the list

#24: Delete all Authenticated Devices from the authenticated device list

Command	Return	Argument
AT+RMSAD	OK	None

#25: Locate Authenticated Device from the authenticated device list

Command	Return	Argument
AT+FSAD=<Param>	1. OK - exists 2. FAIL- no-exisit	Param: Bluetooth device address

Example:
Finddevice with address: 12:34:56:ab:cd:ef
at+FSAD=1234:56:abcdef\r\n
OK

Or
at+fsad=1234:56:abcdef\r\n
FAIL ==== there is no such device in the list

#26: Obtain the total Authenticated Device number in the authenticated device list

Command	Return	Argument
AT+ADCN?=<Param>	+ADCN:<Param> OK	Param: total number of device in the authenticated device list

#27: Obtain the most recently used Authenticated Device

Command	Return	Argument
AT+MRAD?	+MRAD:<Param>	Param: most recently used authenticated device

#28: Obtain the working status of the Bluetooth device

Command	Return	Argument
AT+STATE?	+STATE:<Param> OK	Param: working status “INITIALIZED” “READY” “PAIRABLE” “PAIRD” “INQUIRING” “CONNECTING” “CONNECTED” “DISCONNECTED” “NUKNOW”

#29: Initialise the spp profile lib

Command	Return	Argument
AT+INIT	1. OK 2. FAIL	NONE

#30: Inquire nearby devices

Command	Return	Argument
AT+INQ	+INQ: <Param1>,<Param2>,<Param3> …. OK	Param1: address Param2: device class Param3: RSSI

Example 1:

at+init\r\n —— Initialize SPP (can't repeatedly initialize)
OK
at+iac=9e8b33\r\n ——inquire general inquire access code
OK
at+class=0\r\n —— inquire all devices types
OK
at+inqm=1,9,48\r\n —— Inquire mode: RSSI, max number 9, timeout 48
At+inq\r\n —— inquire
+INQ:2:72:D2224,3E0104,FFBC
+INQ:1234:56:0,1F1F,FFC1
+INQ:1234:56:0,1F1F,FFC0
+INQ:1234:56:0,1F1F,FFC1
+INQ:2:72:D2224,3E0104,FFAD
+INQ:1234:56:0,1F1F,FFBE
+INQ:1234:56:0,1F1F,FFC2
+INQ:1234:56:0,1F1F,FFBE
+INQ:2:72:D2224,3E0104,FFBC
OK

#31: Cancel Inquire nearby devices

Command	Return	Argument
AT+INQC	OK	None

#32: Device pairing

Command	Return	Argument
AT+PAIR=<Param1>,<Param2>	1. OK 2. FAIL	Param1: remote device address Param2:timeout

Example:
Pair with remote device: 12:34:56:ab:cd:ef, timeout 20 s.
At+pair=1234,56,abcdef, 20\r\n
OK

#33: Device Connection

Command	Return	Argument
AT+LINK=<Param>	1. OK 2. FAIL	Param: remote device address

Example:
Link to remote device: 12:34:56:ab:cd:ef
At+fsad=1234,56,abcdef\r\n -- check if remote device is in the authenticated device list or not
OK
At+link==1234,56,abcdef\r\n -- it is in the list, doesn't need to be inquired and can be directly linked
OK
#34: Device Disconnection

Command	Return	Argument
AT+DISC	1. +DISC: SUCCESS 2. +DISC:LINK_LOSS 3. +DISC:NO_SLC 4. +DISC:TIMEOUT 5. +DICS:ERROR	None

#35: Enter into energy saving mode

Command	Return	Argument
AT+ENSNIFF=<Param>	OK	Param: Bluetooth device address

#36: Exit energy saving mode

Command	Return	Argument
AT+EXSNIFF=<Param>	OK	Param: Bluetooth device address

Appendix 1: AT command error

ERROR code decoder

Error_code (hex)	Explanation
0	AT command error
1	The result is default value
2	PSKEY write error
3	Device name is too long (more than 32 bytes)
4	Device name is 0 byte
5	Bluetooth address: NAP is too long
6	Bluetooth address: UAP is too long
7	Bluetooth address: LAP is too long
8	PIO port mask length is 0
9	Invalid PIO port
A	Device class is 0 byte
B	Device class is too long
C	Inquire Access Code length is 0
D	Inquire Access Code is too long
E	Invalid Inquire Access Code
F	Pairing password is 0
10	Pairing password is too long (more than 16 bytes)
11	Role of module is invalid
12	Baud rate is invalid
13	Stop bit is invalid
14	Parity bit is invalid
15	No device in the pairing list
16	SPP is not initialized
17	SPP is repeatedly initialized
18	Invalid inquiry mode
19	Inquiry timeout
1A	Address is 0
1B	Invalid security mode
1C	Invalid encryption mode

Appendix 2: Device Class

The Class of Device/Service(CoD)is a 32 bits number that is made of 3 fields. One field specifies the service supported by the device. Another field specifies the major device class, which broadly corresponds to the type of the device. The third field specifies the minor device class, which describes the device type in more detail.

The Class of Device/Service (CoD) field has a variable format. The format is indicated using the 'Format Type field' within the CoD. The length of the Format Type field is variable and ends with two bits different from '11'. The version field starts at the least significant bit of the CoD and may extend upwards. In the 'format #1' of the CoD (Format Type field = 00), 11 bits are assigned as a bit-mask (multiple bits can be set) each bit corresponding to a high level generic category of service class. Currently 7 categories are defined. These are primarily of a 'public service' nature. The remaining 11 bits are used to indicate device type category and other device-specific characteristics.Any reserved but otherwise unassigned bits, such as in the Major Service Class field, should be set to 0.

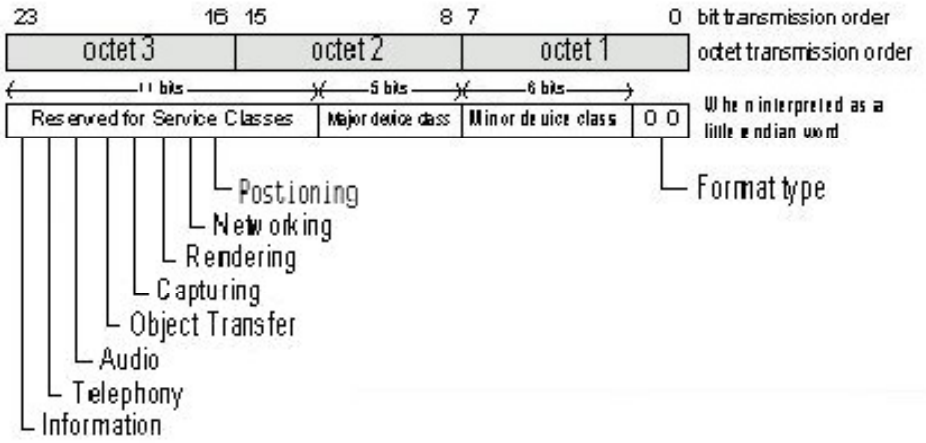


Figure 1.2: The Class of Device/Service field (first format type). Please note the order in which the octets are sent on the air and stored in memory. Bit number 0 is sent first on the air.

1. MAJOR SERVICE CLASSES

Bit no	Major Service Class
13	Limited Discoverable Mode [Ref #1]
14	(reserved)
15	(reserved)
16	Positioning (Location identification)
17	Networking (LAN, Ad hoc, ...)
18	Rendering (Printing, Speaker, ...)
19	Capturing (Scanner, Microphone, ...)
20	Object Transfer (v-Inbox, v-Folder, ...)
21	Audio (Speaker, Microphone, Headset service, ...)
22	Telephony (Cordless telephony, Modem, Headset service, ...)
23	Information (WEB-server, WAP-server, ...)

TABLE 1.2: MAJOR SERVICE CLASSES

[Ref #1 As defined in See Generic Access Profile, Bluetooth SIG]

2. MAJOR DEVICE CLASSES

The Major Class segment is the highest level of granularity for defining a Bluetooth Device. The main function of a device is used to determine the major class grouping. There are 32 different possible major classes. The assignment of this Major Class field is defined in Table 1.3.

12	11	10	9	8	Major Device Class
0	0	0	0	0	Miscellaneous [Ref #2]
0	0	0	0	1	Computer (desktop,notebook, PDA, organizers, ...)
0	0	0	1	0	Phone (cellular, cordless, payphone, modem, ...)
0	0	0	1	1	LAN /Network Access point
0	0	1	0	0	Audio/Video (headset,speaker,stereo, video display, vcr.....
0	0	1	0	1	Peripheral (mouse, joystick, keyboards, ...)
0	0	1	1	0	Imaging (printing, scanner, camera, display, ...)
1	1	1	1	1	Uncategorized, specific device code not specified
X	X	X	X	X	All other values reserved

TABLE 1.3: MAJOR DEVICE CLASSES

[Ref #2: Used where a more specific Major Device Class code is not suited (but only as specified in this document). Devices that do not have a major class code assigned can use the all-1 code until 'classified']

3. THE MINOR DEVICE CLASS FIELD

The 'Minor Device Class field' (bits 7 to 2 in the CoD), are to be interpreted only in the context of the Major Device Class (but independent of the Service Class field). Thus the meaning of the bits may change, depending on the value of the 'Major Device Class field'. When the Minor Device Class field indicates a device class, then the primary device class should be reported, e.g. a cellular phone that can also work as a cordless handset should use 'Cellular' in the minor device class field.

4. MINOR DEVICE CLASS FIELD - COMPUTER MAJOR CLASS

7	6	5	4	3	2	Minor Device Class bit no of CoD
0	0	0	0	0	0	Uncategorized, code for device not assigned
0	0	0	0	0	1	Desktop workstation
0	0	0	0	1	0	Server-class computer
0	0	0	0	1	1	Laptop
0	0	0	1	0	0	Handheld PC/PDA (clam shell)
0	0	0	1	0	1	Palm sized PC/PDA
0	0	0	1	1	0	Wearable computer (Watch sized)
X	X	X	X	X	X	All other values reserved

TABLE 1.4: SUB DEVICE CLASS FIELD FOR THE 'COMPUTER' MAJOR CLASS

5. MINOR DEVICE CLASS FIELD - PHONE MAJOR CLASS

7	6	5	4	3	2	Minor Device Class bit no of CoD
0	0	0	0	0	0	Uncategorized, code for device not assigned
0	0	0	0	0	1	Cellular
0	0	0	0	1	0	Cordless
0	0	0	0	1	1	Smart phone
0	0	0	1	0	0	Wired modem or voice gateway
0	0	0	1	0	1	Common ISDN Access
0	0	0	1	1	0	Sim Card Reader
X	X	X	X	X	X	All other values reserved

6. MINOR DEVICE CLASS FIELD - LAN/NETWORK ACCESS POINT MAJOR CLASS

7	6	5	Minor Device Class bit no of CoD
0	0	0	Fully available
0	0	1	1 - 17% utilized
0	1	0	17 - 33% utilized
0	1	1	33 - 50% utilized
1	0	0	50 - 67% utilized
1	0	1	67 - 83% utilized

1	1	0	83 - 99% utilized
1	1	1	No service available [REF #3]
X	X	X	All other values reserved

TABLE 1.6: THE LAN/NETWORK ACCESS POINT LOAD FACTOR FIELD

[Ref #3: "Device is fully utilized and cannot accept additional connections at this time, please retry later"]
The exact loading formula is not standardized. It is up to each LAN/Network Access Point implementation to determine what internal conditions to report as a utilization percentage. The only requirement is that the number reflects an ever-increasing utilization of communication resources within the box. As a recommendation, a client that locates multiple LAN/Network Access Points should attempt to connect to the one reporting the lowest load.

4	3	2	Minor Device Class bit no of CoD
0	0	0	Uncategorized (use this value if no other apply)
X	X	X	All other values reserved

TABLE 1.7: RESERVED SUB-FIELD FOR THE LAN/NETWORK ACCESS POINT

7. MINOR DEVICE CLASS FIELD - AUDIO/VIDEO MAJOR CLASS

7	6	5	4	3	2	Minor Device Class bit no of CoD
0	0	0	0	0	0	Uncategorized, code for device not assigned
0	0	0	0	0	1	Device conforms to the Headset profile
0	0	0	0	1	0	Hands-free
0	0	0	0	1	1	(Reserved)
0	0	0	1	0	0	Microphone
0	0	0	1	0	1	Loudspeaker
0	0	0	1	1	0	Headphones
0	0	0	1	1	1	Portable Audio
0	0	1	0	0	0	Car audio
0	0	1	0	0	1	Set-top box
0	0	1	0	1	0	HiFi Audio Device
0	0	1	0	1	1	VCR
0	0	1	1	0	0	Video Camera
0	0	1	1	0	1	Camcorder
0	0	1	1	1	0	Video Monitor
0	0	1	1	1	1	Video Display and Loudspeaker
0	1	0	0	0	0	Video Conferencing
0	1	0	0	0	1	(Reserved)

0	1	0	0	1	0	Gaming/Toy [Ref #4]
X	X	X	X	X	X	All other values reserved

[Ref #4: Only to be used with a Gaming/Toy device that makes audio/video capabilities available via Bluetooth]

TABLE 1.8: SUB DEVICE CLASSES FOR THE 'AUDIO/VIDEO' MAJOR CLASS

8. MINOR DEVICE CLASS FIELD - PERIPHERAL MAJOR CLASS

7	6	Minor Device Class bit no of CoD
0	1	Keyboard
1	0	Pointing device
1	1	Combo keyboard/pointing device
X	X	All other values reserved

TABLE 1.9: THE PERIPHERAL MAJOR CLASS KEYBOARD/POINTING DEVICE FIELD

Bits 6 and 7 independently specify mouse, keyboard or combo mouse/keyboard devices. These may be combined with the lower bits in a multifunctional device.

5	4	3	2	Minor Device Class bit no of CoD
0	0	0	0	Uncategorized device
0	0	0	1	Joystick
0	0	1	0	Gamepad
0	0	1	1	Remote control
0	1	0	0	Sensing device
0	1	0	1	Digitizer tablet
X	X	X	X	All other values reserved

TABLE 1.10: RESERVED SUB-FIELD FOR THE DEVICE TYPE

9. MINOR DEVICE CLASS FIELD - IMAGING MAJOR CLASS

7	6	5	4	Minor Device Class bit no of CoD
X	X	X	1	Display
X	X	1	X	Camera
X	1	X	X	Scanner
1	X	X	X	Printer
X	X	X	X	All other values reserved

TABLE 1.11: THE IMAGING MAJOR CLASS BITS 4 TO 7

Bits 4 to 7 independantly specify display, camera, scanner or printer. These may be combined in a multifunctional device.

3	2	Minor Device Class bit no of CoD
0	0	Uncategorized, default
X	X	All other values reserved

TABLE 1.12: THE IMAGING MAJOR CLASS BITS 2 AND 3

Bits 2 and 3 are reserved

Appendix 3 The Inquiry Access Codes

The General- and Device-Specific Inquiry Access Codes (DIACs)

The Inquiry Access Code is the first level of filtering when finding Bluetooth devices and services. The main purpose of defining multiple IACs is to limit the number of responses that are received when scanning devices within range.

0. 0x9E8B33 —— General/Unlimited Inquiry Access Code (GIAC)

1. 0x9E8B00 —— Limited Dedicated Inquiry Access Code (LIAC)

2. 0x9E8B01 ～ 0x9E8B32 RESERVED FOR FUTURE USE

3. 0x9E8B34 ～ 0x9E8B3F RESERVED FOR FUTURE USE

The Limited Inquiry Access Code (LIAC) is only intended to be used for limited time periods in scenarios where both sides have been explicitly caused to enter this state, usually by user action. For further explanation of the use of the LIAC, please refer to the Generic Access profile.

In contrast it is allowed to be continuously scanning for the General Inquiry Access Code (GIAC) and respond whenever inquired.