

Sociable Objects Workshop

Instructor: Rob Faludi

Plan for Today

- Simple Sensor Network: presentation and demo
- Collaboration
- I/O API review and Remote AT commands
- Sleep Mode
- End Devices
- Direct Actuation
- Readings & Assignments

Simple Sensor Network

Collaboration

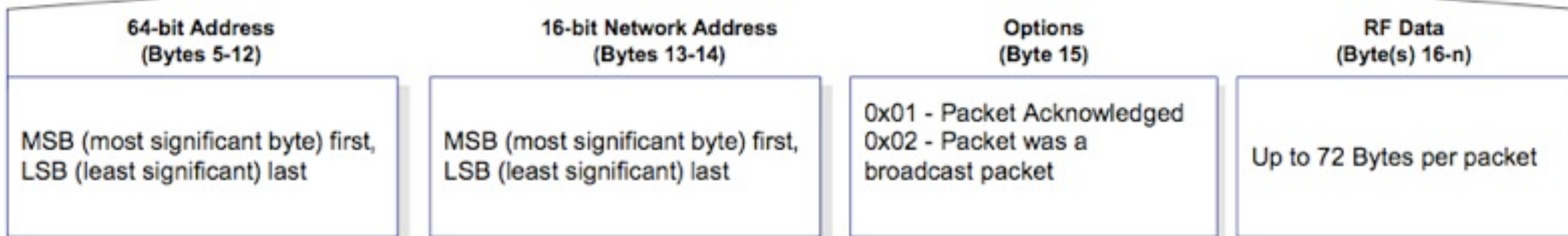
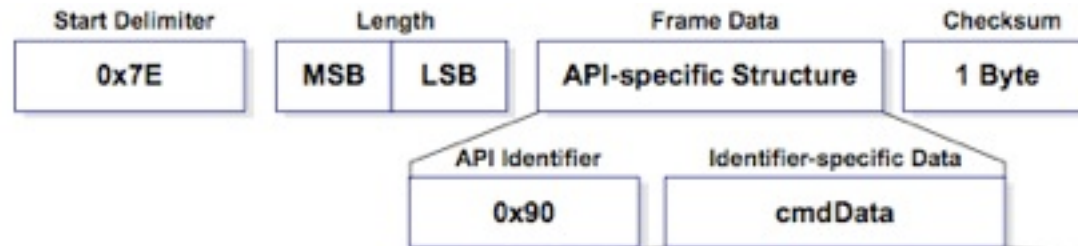
Collaboration

- more information
- multiple alternatives
- diversity of backgrounds
- common goals
- balance of power
- no forced consensus

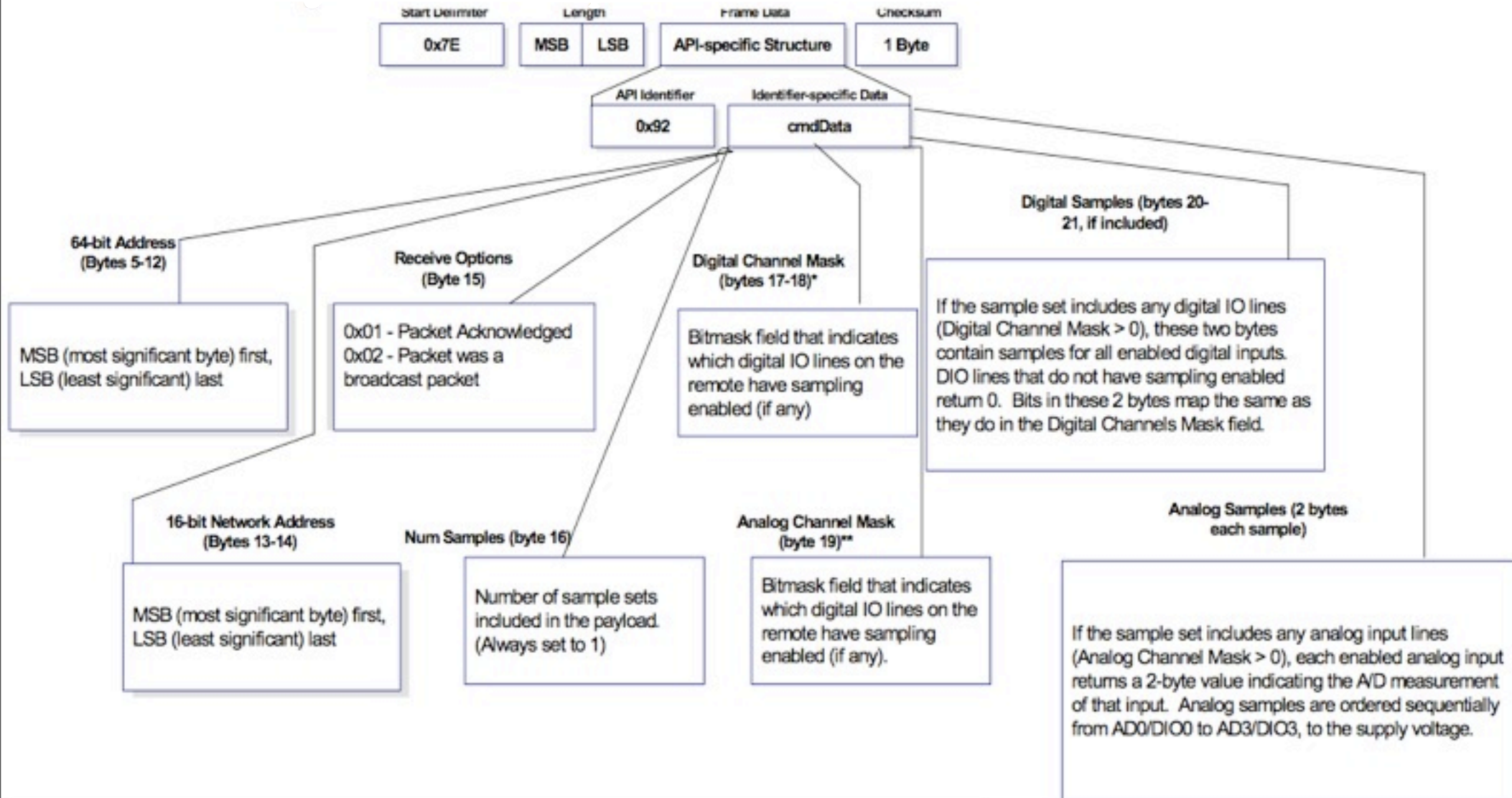
API for I/O and Remote AT

RX Packet

- Maximum of 72 bytes of data per packet
- RF Data section is basis for I/O packets



I/O RX Packet



I/O Digital Channel Mask and Digital Data

Digital Channel Mask (bytes 17-18)*

Bitmask field that indicates which digital IO lines on the remote have sampling enabled (if any)

*

N/A	N/A	N/A	CD/DIO 12	PWM/DI O11	RSSI/DI O10	N/A	N/A
CTS/DI O7	RTS/DI O6	ASSOC/ DIO5	DIO4	AD3/DI O3	AD2/DI O2	AD1/DI O1	AD0/DI O0

Digital Samples (bytes 20- 21, if included)

If the sample set includes any digital IO lines (Digital Channel Mask > 0), these two bytes contain samples for all enabled digital inputs. DIO lines that do not have sampling enabled return 0. Bits in these 2 bytes map the same as they do in the Digital Channels Mask field.

I/O Analog Channel Mask and Analog Samples

Analog Channel Mask (byte 19)**

Bitmask field that indicates which digital IO lines on the remote have sampling enabled (if any).

**

Supply Voltage	N/A	N/A	N/A	AD3	AD2	AD1	AD0
----------------	-----	-----	-----	-----	-----	-----	-----

Analog Samples (2 bytes each sample)

If the sample set includes any analog input lines (Analog Channel Mask > 0), each enabled analog input returns a 2-byte value indicating the A/D measurement of that input. Analog samples are ordered sequentially from AD0/DIO0 to AD3/DIO3, to the supply voltage.

I/O Structure Reviewed

- Num Samples (1 byte)
 - Digital Channel Mask (2 bytes)
 - Analog Channel Mask (1 byte)
 - Two bytes of digital data IF ANY DIGITAL CHANNELS ENABLED followed by...
 - ...two bytes for EACH analog channel enabled...
-
- Q: How many bytes ATD02 ATD12 ATD23?

I/O Bytes Example

0x7E (start byte)

0x00

0x17 (length)

0x92 (API id)

0x00 (64-bit address)

0x13

0x20

0x00

0x43

0x23

0x12

0xEF

0x03 (16-bit address)

0xA4

0x01 (num samples)

0x00 (digital channel masks)

0x00

0x01 (analog channel mask)

0x02 (first analog sample)

0xF8

0x30 (the checksum)

I/O Code: Basic

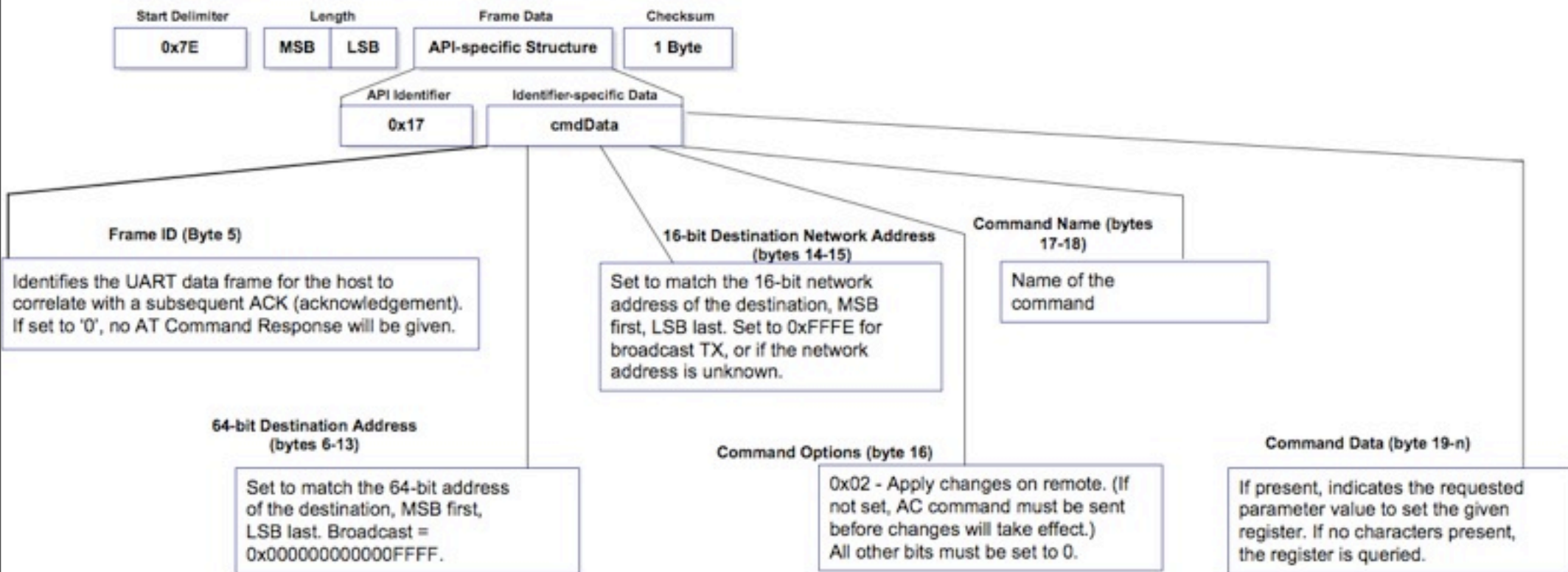
- Fixed parameters make for easier programming
- Assume we are just reading a single ADC channel:

Arduino Version:

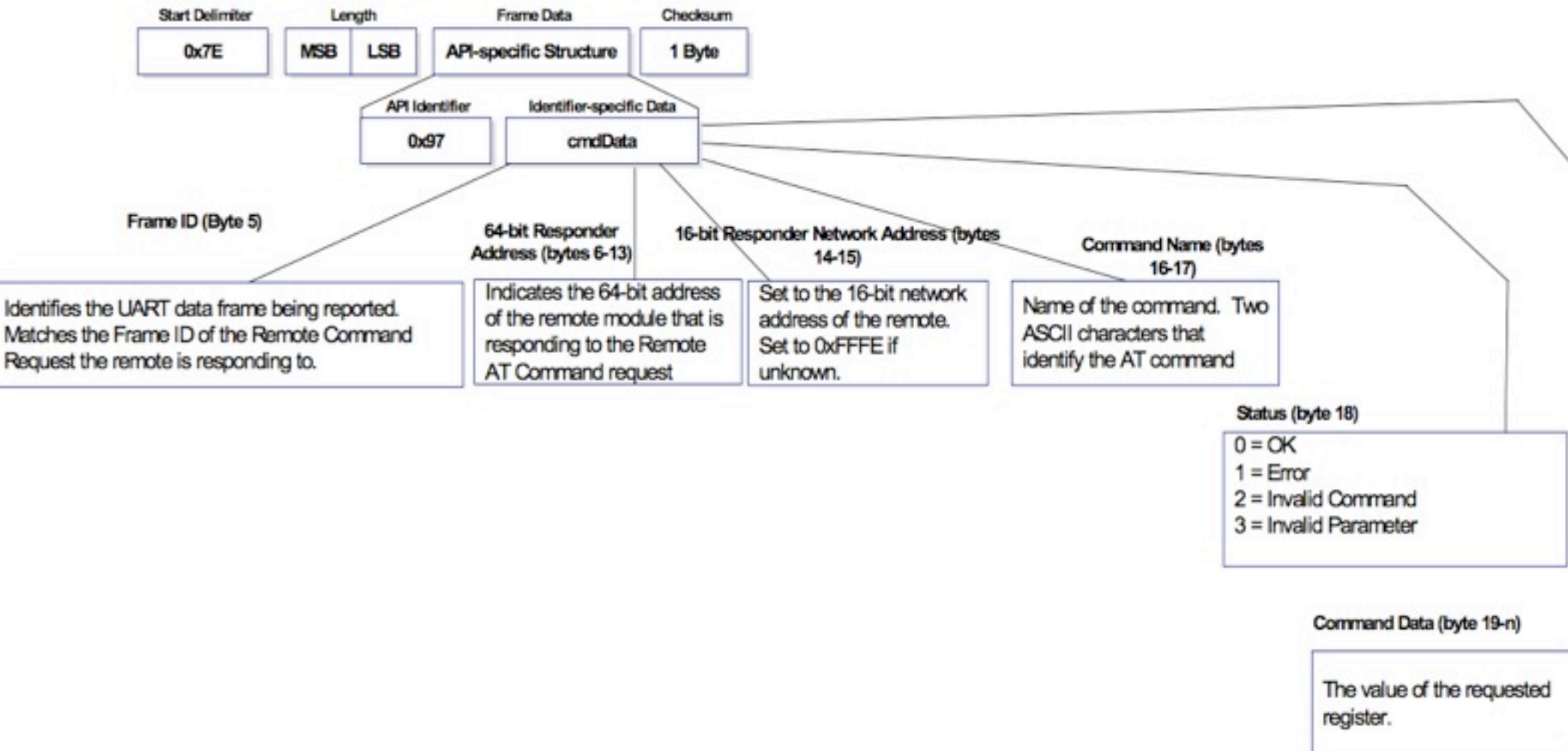
```
// make sure everything we need is in the buffer
if (Serial.available() >= 21) {
  // look for the start byte
  if (Serial.read() == 0x7E) {
    // read the variables that we're not using out of the buffer
    for (int i = 0; i<18; i++) {
      byte discard = Serial.read();
    }
    int analogHigh = Serial.read();
    int analogLow = Serial.read();
    analogValue = analogLow + (analogHigh * 256);
  }
}
```

Remote AT Command Request

- Send commands across the network



Remote AT Command Response



Remote AT Command Code

```
void setRemoteState(int value) { // pass either a 0x4 or and 0x5 to turn the pin on or off
  Serial.print(0x7E, BYTE);
  Serial.print(0x0, BYTE); // high part of length (always zero)
  Serial.print(0x10, BYTE); // low part of length (the number of bytes that follow, not including checksum)
  Serial.print(0x17, BYTE); // 0x17 is a remote AT command
  Serial.print(0x0, BYTE); // frame id set to zero for no reply
  // ID of recipient, or use 0xFFFF for broadcast
  Serial.print(00, BYTE);
  Serial.print(00, BYTE);
  Serial.print(00, BYTE);
  Serial.print(00, BYTE);
  Serial.print(00, BYTE);
  Serial.print(00, BYTE);
  Serial.print(0xFF, BYTE); // 0xFF for broadcast
  Serial.print(0xFF, BYTE); // 0xFF for broadcast
  // 16 bit of recipient or 0xFFFE if unknown
  Serial.print(0xFF, BYTE);
  Serial.print(0xFE, BYTE);
  Serial.print(0x02, BYTE); // 0x02 to apply changes immediately on remote
  // command name in ASCII characters
  Serial.print('D', BYTE);
  Serial.print('1', BYTE);
  // command data in as many bytes as needed
  Serial.print(value, BYTE);
  // checksum
  long sum = 0x17 + 0xFF + 0xFF + 0xFF + 0xFE + 0x02 + 'D' + '1' + value;
  Serial.print( 0xFF - ( sum & 0xFF) , BYTE );
  delay(10);
}
```


End Devices

ZigBee End Device

- Optional to ZigBee networks
- Typically battery-powered
- Many can be on each PAN
- Issues a beacon request on startup to locate channel and PAN
- Automatically attempts to join a valid PAN
- End devices can only communicate directly with their parent
- 16-bit address assigned by coordinator

Sleep Mode

Sleeping the XBee ZigBee: Basics

- Why Sleep?
- ATSM
 - 1: pin hibernate, $<10 \mu\text{A}$, 13.2 ms wakeup, uses pin 9
 - 2 and 3: <nothing>
 - 4: cyclic sleep, also $<50 \mu\text{A}$, 2 ms wakeup, module must be idle
 - 5: cyclic sleep with pin wakeup
- ATSP: Sleep Period (* 10 ms) [0x20-0xAFO]
- ATSN: Number of Sleep Periods (* 1 ms)
used in external device control
- ATST: Sleep Timer (* 1 ms)
- ATSO: Options 0x2 = regular, 0x4 = extended



Sleep Commands

Table 10-015.Sleep Commands

AT Command	Name and Description	Node Type ¹	Parameter Range	Default
SM	Sleep Mode Sets the sleep mode on the RF module	E	0-Sleep disabled 1-Pin sleep enabled 4-Cyclic sleep enabled 5 - Cyclic sleep, pin wake	0
SN	Number of Sleep Periods. Sets the number of sleep periods to not assert the On/Sleep pin on wakeup if no RF data is waiting for the end device. This command allows a host application to sleep for an extended time if no RF data is present	CRE	1 - 0xFFFF	1
SP	Sleep Period. This value determines how long the end device will sleep at a time, up to 28 seconds. (The sleep time can effectively be extended past 28 seconds using the SN command.) On the parent, this value determines how long the parent will buffer a message for the sleeping end device. It should be set at least equal to the longest SP time of any child end device.	CRE	0x20 - 0xAF0 x 10ms (Quarter second resolution)	0x20
ST	Time Before Sleep Sets the time before sleep timer on an end device.The timer is reset each time serial or RF data is received. Once the timer expires, an end device may enter low power operation. Applicable for cyclic sleep end devices only.	E	1 - 0xFFFFE (x 1ms)	0x1388 (5 seconds)
SO Command	Sleep Options. Configure options for sleep. Unused option bits should be set to 0. Sleep options include: 0x02 - Always wake for ST time 0x04 - Sleep entire SN * SP time Sleep options should not be used for most applications. See Sleep Mode chapter for more information.	E	0 - 0xFF	0
WH	Wake Host. Set/Read the wake host timer value. If the wake host timer is set to a non-zero value, this timer specifies a time (in millisecond units) that the device should allow after waking from sleep before sending data out the UART or transmitting an IO sample. If serial characters are received, the WH timer is stopped immediately.	E	0 - 0xFFFF (x 1ms)	

Sleeping the XBee Zigbee: Example

- ATSM5,SP64
 - Will wake up on pin 9 high, and also every 1000 ms
- Use in conjunction with I/O readings
 - Wakeup will always trigger an I/O sample
 - More samples if ATIR allows it during the awake period
- Everything in the PAN should have SP set the same or greater so that incoming frames for the end devices are buffered properly



Direct Actuation

Direct Actuation

- Direct output, i.e. without a microcontroller, should be used sparingly
 - no logic
 - limited pins
 - no feedback
 - API packets have to be perfect
- Only if you need to save money on a large number of devices or need your device to be as small as possible. Otherwise get a μ -controller!

Readings and Assignments

- Readings Watch a Movie!
 - Fast, Cheap and Out of Control
 - in the ER, on iPhone or Netflix
- Assignments
 - Sensor/Actuator Proposal(s)