“The Spirit of USB”, Part II

Electronics and (target) device programming

Robert Gyllenberg,
Jerker Björkqvist,
2005 Åbo Akademi

Contents

- Signals and encoding
- Electrical interface
- Power management
- Client (target) side programming
- Practical demonstation

Contents

- Signals and encoding
  - Bus states
  - Data encoding
  - Packet format
  - Test modes
- Electrical interface
- Power management
- Client (target) side programming
- Practical demonstation

Bus States (low and full speed)

- Electrical states and corresponding levels
  - Differential 0
  - Differential 1
  - Single-ended 0
  - Single-ended 1
  - High-Z

Bus States (low and full speed)

- Logical states
  - Data J
  - Data K
  - Idle
  - Resume
  - Start-of-Packet (SOP)
  - End-of-Packet (EOP)
  - Disconnect
  - Connect
  - Reset

Bus States (high speed)

- Electrical states and corresponding levels
  - high-speed differential 0
  - high-speed differential 1

Bus States (high speed)

- Logical states
  - high-speed data J
  - high-speed data K
  - chirp J
  - chirp K
  - high-speed Squelch
  - high-speed Idle
  - start of high-speed Packet
  - end of high-speed Packet
  - high-speed Disconnect

Data Encoding

- Non-Return to Zero inverted (NRZI) with bit stuffing
- Encoding keeps receiver synchronized on bit level
- No need for start och stop bits
- Better throughput
- Bit stuffing ensures transitions regardless of data input (0.8% overhead in average, max 17%)

Data Encoding

- Synchronazation fields
  - 8-bits: 'KJKJKJKK' for low and full speed packets
  - 32-bits: 'KJKJKJKJKJKJKJKJKJKJKJKJKJKJKJKK' for high speed packets
- End-of-Packet (EOP)
  - Single-ended-Zero for low and full speed packets
  - More complicated formula for high-speed packets

Timing accuracy

- Deviation allowed in devices
  - 0.05% (500 ppm) for 480 Mbps
  - 0.25% (250 ppm) for 12 Mbps
  - 1.5% (15.000 ppm) for 1.5 Mbps
- Deviation at USB 2.0 hubs
  - data rate within 0.05% (500 ppm)
  - frame interval within 1 ms +/- 500 ns
  - microframe interval 125.0 µs +/- 62.5 µs

Packet format

- All USB data travel in packets, which are blocks of information with a defined format
- Field types
  - SYNC (at beginning of each packet, 8 or 32-bit)
  - PID (packet type identifier, 4+4 -bit)
  - Address (device address, 7-bit)
  - Endpoint (endpoint number within device, 4-bit)
  - Frame Number (identifies the frame, 11-bit)
  - Data (payload, 0-1024 Bytes)
  - CRC (for error detection, 5 or 16-bit)

Packet format

- SYNC (at beginning of each packet)
  - 8 bit 'KJKJKJKK'
  - 32 bit 'KJKJKJKJKJKJKJKJKJKJKJKJKJKJKJKK'

Packet format

- PID (packet type identifier, 4+4 -bit)
  - Bits 0-3 identify the type of packet
  - Bits 4-7 are the one's complement of bits 0-3
  - One of the 16 predefined types
    - token, data, handshake and special packets
    - the two lower bits identify the PID type
    - the two upper bits identify the specific PID

Packet format

- Address (device address, 7-bit)
  - The device number that the host is communicating with.
  - 7-bit -> 127 different numbers
- Endpoint (endpoint address, 4-bit)
  - The endpoint within the addressed device
  - 4-bits -> maximum of 16 endpoints/device

Packet format

- Frame Number (identifies the frame, 11-bit)
  - Sequence number transmitted in each Start-of-Frame packet that begins a frame or microframe.
  - After 0x7FF, the number rolls over to zero
  - Full-speed hosts count each frame (11-bit)
  - High-speed hosts count each microframe (14-bit),but only bits 3-13 are transmitted
    - -> eight microframes have the same frame number

Packet format

- Data (payload, 0-1024 Bytes)
  - The data field ranges from 0 to 1024 bytes, depending on transfer type, bus speed and the amount of data.
- CRC (Cyclic Redundancy Check)
  - 5-bit for address- and endpoint fields and 16-bit for data fields
  - CRC is taken care of by hardware in both directions

Inter-packet Delay

- Half duplex nature of USB
  - data can only travel in one direction at a time.
  - the previous transmitting driver must switch off, before the next transmitter can turn its driver on
  - a brief delay is required between packets
  - different specs for different speeds
  - taken care of by hardware

Contents

- Signals and encoding
- Electrical interface
  - Cables and connectors
  - Transceivers and signals
  - Ensuring signal quality
- Power management
- Client (target) side programming
- Practical demonstation

Contents

- Signals and encoding
- Electrical interface
  - Cables and connectors
  - Transceivers and signals
  - Signal voltages
  - Ensuring signal quality
- Power management
- Client (target) side programming
- Practical demonstation

Cables

- Supply voltage lines
  - VBUS (+5V), GND (0V)
  - Non-twisted
  - #28.. #20 AWG (x..y mm^2)

Cables

- Data lines
  - D+ and D-
  - Shielding
    - Double shield with drain wire required in USB 2.0
    - Double shield also *recommended* for low-speed, but single inner shield and drain wire can do for low-speed devices
  - Electrical impedance
    - Common mode 30 ohms +/- 30%
    - Differential mode 90 ohms

Connectors

- The USB-specification also defines the connectors to be used and their pinouts
  - four plug types
    - A, B, mini-A, mini-B
  - four mating receptacle types

Connectors

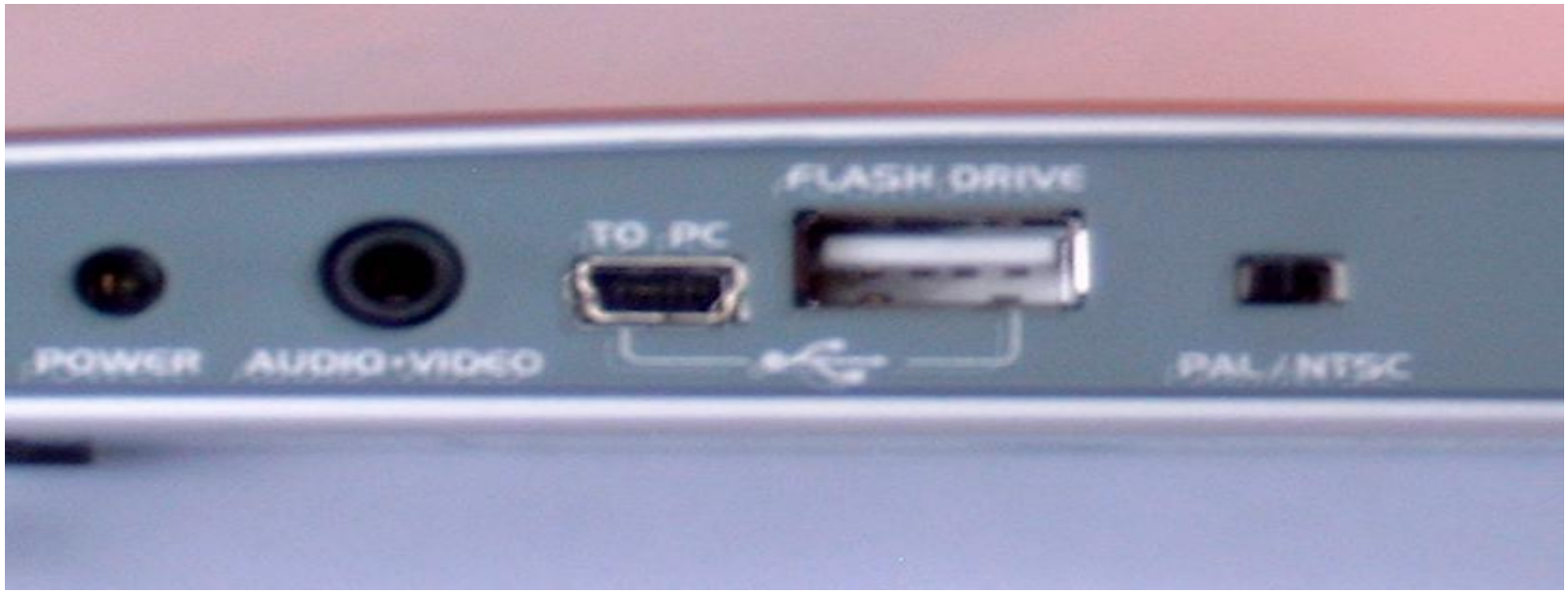
- Series A
  - for upstream end of the cable
  - some small devices integrated with the plug
    - memory sticks
    - music players
    - etc.

Connectors

- Series B (standard and miniature)
  - for downstream end of the cable
  - can be replaced by mini-B, if standard B is too bulky
  - may be left out (hard wired or connected with non-standard USB connector)

Connectors

- Miniature A
  - added by On-the-Go supplement for such hosts



Connectors

- Down stream end of cable
  - Detachable cable
    - Has standard or miniature B-series connector
    - Must support full and high speed USB
  - Captive cable
    - Permanently connected
    - or still removable **using non-standard connector**
  - **Low-speed cables shall not physically interconnect with high-speed devices**

Pinout and cable colours

Series A or series B pin	Mini-B pin	Conductor	Cable Wire
1	1	VBUS	red
2	2	D-	white
3	3	D+	green
4	5	GND	black
-	4	ID	n.c.
shell		shield	drain wire

Transceivers and Signals

- DC properties
  - Nominal supply: 5 V, max. 500 mA
  - Data levels: -0.01 V .. + 3.6 V
  - Short proof
    - Must withstand shortage between any of the four wires without damage!

Transceivers and signals

- Electrical information
  - Voltage levels
  - Resistance
- Appearance
  - Differential
  - Common mode
  - Single-ended (on one or both wires)

Transceivers and Signals

- AC properties vary with speed
  - Different edge rates (-> different rise and fall times)
  - Different supporting circuits needed

Transceivers and Signals

- AC properties vary with speed
  - Different edge rates (-> different rise and fall times)
    - Low-speed devices have longer rise and fall times thand full or high-speed devices.
    - Makes use of cheeper (un-twisted) cable possible
      - Good for e.g. mice
  - Different supporting circuits needed

Transceivers and Signals

- AC properties vary with speed
  - Different edge rates (-> different rise and fall times)
  - Different supporting circuits needed
    - Pull-up resistance on D+ or D- used to identify low- or full speed devices
    - Differential termination only (no pullup) identifies presence of a high-speed device

Ensuring signal quality

- Cable length
- Sources of noise
- Balanced lines
- Twisted pairs
- Shielding
- Edge rates
- Isolated interfaces

Contents

- Signals and encoding
- Electrical interface
- Power management
  - Powering Options
  - Hub Power
  - Saving Power
- Client (target) side programming
- Practical demonstation

Managing power

- USB offers the ability for devices to draw power from the bus.
- Many devices entirely bus powered
- Bus Power is, however, a limited resource
- Device design carries the responsibility to live within the limits of available power
- Important design choice: to use bus power or not?

Powering options

- Drawing power from the USB (PC or hub) is advantageous:
  - No need for an electrical outlet and a "wall bug" for each peripheral
  - Devices become lighter and cheaper
  - No need for electrical safety precations regarding the power supply
- Disadvantage:
  - Most self-powered USB-hubs use "wall bugs"

Powering options

- Drawing power from a PC isn't new
  - none of the traditional serial (RS-232) or parallell (Centronics) ports support *dedicated* powering, but data lines can be *misused* for small powering needs.
  - some former connectors (e.g. for joystick or keyboard) have dedicated power supply pins
- With USB, you don't have to resort to any powering tricks
  - the USB-specification allows if "officially"

Voltages

- Depending on the hub type, the voltage ranges are:
  - 4.4 to 5.25 V (low power hub)
  - 4.75 to 5.25 V (high power hub)
- Transient contitions
  - can drop the voltage to as low as 4.07 V

Voltages

- Voltage can drop the voltage to as low as 4.07 V
  - Devices that need full +5V supply can use an internal step-up converter
  - Devices that need only +3.3V or less can use step-down or a low dropout linear regulator

Which peripherals can use power from the USB bus?

- All devices that draw less than 100 mA and only need power when attached to the bus
- Devices that draw a maximum of 500 mA when active, but less than 100 mA when 'asleep' or during configuration and are not operated by battery powered hosts

Which peripherals can NOT use power from the USB bus?

- Any device that needs more than 500 mA
- Any device that needs to function when not attached to the bus
- Any device that need more than 100 mA and is intended to be used with battery powered hosts

USB Power Definitions

- Low-power device
  - draws up to 100 mA
- High-power device
  - draws up to 500 mA
- Self-powered device
  - can draw 0-100 mA from the bus
  - the rest is drawn from the devices own supply

Power during enumeration

- Even a high-power device must be able to enumerate at low power
- On power-up any device may not draw more than 100 mA until the device is configured during the enumeration process
- Hosts and hubs are likely to allocate either 100 mA or 500 mA for a device, rather than the precise amount of **`bMaxPower`**

Important notes

- The current limits are absolute maximum values, not averages.
- Beware of the possibility of increased current in case of +5.25 Volt at the bus supply
- No device may ever provide upstream power
- Even the pull-up resistor must remain unpowered until V_{BUS} is present
- Self powered devices *must have a connection to to detect the V_{BUS} voltage*, even if it is never used!

Informing the host

- According to the USB-specification, each device's configuration descriptor holds a bMaxPower value that specifies the maximum bus current the device may need.
- All hubs have over-current protection that prevents excessive current to flowing to a device.
- The over-current protection circuitry can also inform the OS and the end user about an over-current condition, if it occurs.

Informing the host

- The pull-up resistors and differential termination resistors are used to inform the host about connections and disconnections.
- A device that e.g. changes from bus-power to self-power or vice versa, can toggle the pullup or termination to inform the host about such a change.

Contents

- Signals and encoding
- Electrical interface
- Power management
- Client (target) side programming
  - What is it all about?
  - Getting started
- Practical demonstation

Client side programming

- What is it all about?
  - Reacting to the USB-requests from the host
  - Following the same specification from another point of view

Reacting to the host's requests

- USB-communication always initiated by the host; never by the client.
- Providing the necessary descriptors
- Receiving commands and data
- Providing ”data reports”

Getting started

- Classes & Generic Devices
  - the USB Device Working Group (”DWG”) is responsible for the development of USB Class Specifications for the USB-IF Inc.
  - A single class driver can manage any device that is compatible with a specific class
    - related devices
    - different vendors

Implementing USB Classes

- What is a USB-class?
  - A group of devices with
    - common attributes
    - common services

Class example: Test and Measurement class

- Test and Measurement (USBTMC)
  - Instrumentation class without need for guaranteed timing
  - ADC, DAC, sensors, transducers
  - Bulk IN and Bulk OUT endpoints
  - USB488 is a subclass of USBTMC
    - Interrupt IN for USB488 subclass

Getting started

- Obtain the appropriate starter kit for a platform of your choice.
- Implementing USB from scratch on either host or target side would rather be a waste of time than a good exercise.

Getting started

- Start with a ”skeleton” -application of an appropriate class.
- Add any special functionality needed (and remove any unnecessary or ”dead” code.)

Getting started

- Hands-on practise usually the most efficient method for learning HW/SW-design.
- Participants should participate in laboratory exercises individually or in very small groups (2-3 persons)

What about the Vendor ID ???

- Do I have to invest a lot of mone to obtain a Vendor ID of my own?
  - Basically: Yes, every manufacturer of USB-clients must have a Vendor-ID of his own!
  - In practice: Limited sublicensing possible from e.g. chip manufacturers possible.

Let’s move to the practical part...

- Batch submission of Questions that come up are welcome after both sessions
- Will be answered through e-mail by
  - Jerker Björkqvist
  - Robert Gyllenberg

Case study

- USB starter kit from Microchip
- Comes with the following SW:
  - Host driver pre-compiled binary .DLL
  - Host demo SW binary and source
  - Full client SW binary and source
    - Protocoll stack
    - Demo application