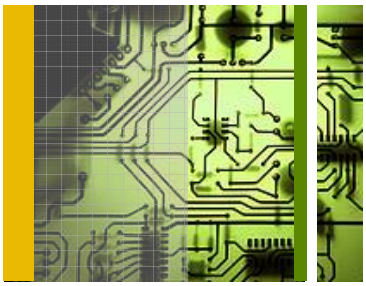
Efficiently Triggering, Debugging and Decoding Low-Speed Serial Buses

Tektronix

Hello, my name is Jerry Mark, I am a Product Marketing Engineer at Tektronix. The following presentation discusses aspects of embedded design techniques for “Efficiently Triggering, Debugging, and Decoding Low-Speed Serial Buses”.

Agenda

- Introduction
  - Parallel Interconnects
  - Transition from Parallel to Serial Buses
  - High-Speed versus Low-Speed Serial Data Buses
- Low-Speed Serial Data Buses
  - Challenges
  - Technology Reviews
  - Measurement Solutions
- Summary

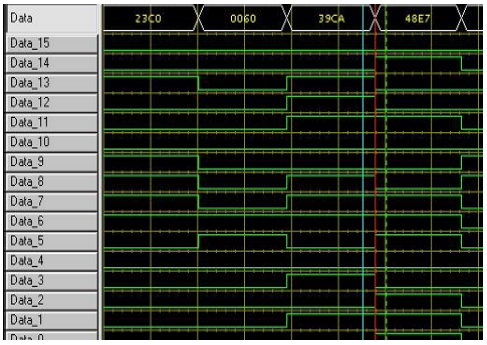


Tektronix

In this presentation we will begin by taking a glance at the transition from parallel-to-serial data and the challenges this presents to engineers. Then we will briefly review some of the most widely used low-speed serial buses in industry today and some of their key characteristics. We will turn our attention to the key measurements on these buses. Subsequently, we will present by example how the low-speed serial solution from Tektronix addresses these challenges.

Parallel Interconnects

- Traditional way to connect digital devices used parallel buses
- Advantages
  - Simple point-to-point connections
  - All signals are transmitted in parallel, simultaneously
  - Easy to capture state of bus (if you have enough channels!)
  - Decoding the bus is relatively easy
- Disadvantages
  - Occupies a lot of circuit board space
  - All high-speed connections must be the same length
  - Many connections limit reliability
  - Connectors may be very large



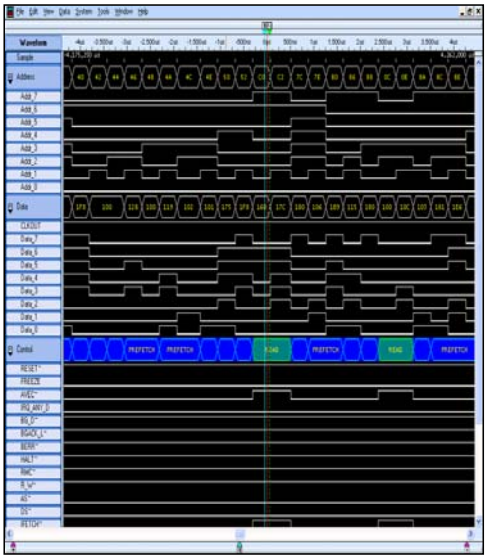
Tektronix

Parallel buses present all of the bits in parallel, as shown in the logic analyzer display. A parallel connection between two ICs can be as simple as point-to-point circuit board traces for each of the data lines. If you can physically probe these lines, it is easy to display the bus state on a logic analyzer or mixed signal oscilloscope. The state of the bus at any point in time is shown vertically on the display. However, for embedded systems, the circuit board space for the parallel buses can make it difficult to meet the product size requirements. For proper operation at high speeds, the length of each of the connections must be carefully matched, making the design more difficult, and perhaps occupying even more circuit board space. The many circuit board connections also limit the reliability of the product, since the solder connections are often the least reliable parts of the circuit. If connectors are required, they tend to be large. Finally, transmission distances for parallel buses are fairly limited and the effects of NEXT (Near Edge Cross Talk) and FEXT (Far Edge Cross Talk) can cause undesired data integrity with multiple parallel traces.

Transition from Parallel to Serial Buses

- Parallel data
  - All data is transferred at the same time and clocked across address and data lines (can be wider than 64 bits)
- Event Triggering
  - State or Pattern event triggering is relatively easy with a logic analyzer or modern oscilloscope
- Decoding
  - Bus decoding is relatively easy, we just look at the state

…this changes with serial!

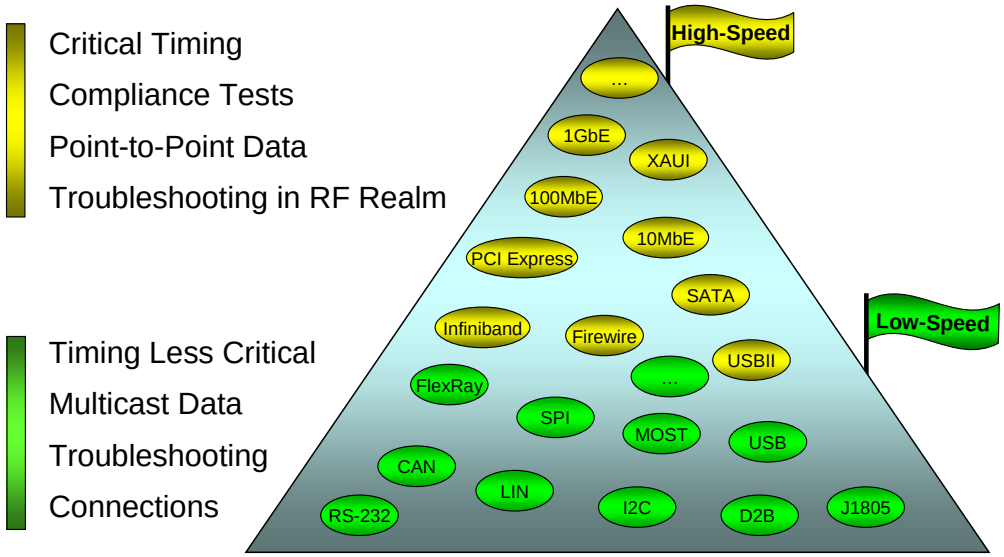


4 |

Tektronix

Transitioning from parallel to serial data buses has been driven by the need to make electronic devices smaller, use less cabling and wires, reduce power, and increase speed to support emerging technologies. In the diagram shown, we can see it is relatively easy to interpret the state of parallel data on buses, based on the clock transferred information, at the same time. Also, State or Pattern event triggering is relatively easy to setup with a logic analyzer or mixed signal oscilloscope. And, reading the parallel data lines in hexadecimal or binary representation is straightforward by examining the logic state of the bus. But all this changes with serial data.

High-Speed Serial versus Low-Speed Serial



Tektronix

Consider the pyramid diagram shown, where we see higher speed serial devices near the top of the pyramid and lower speed devices near the bottom, making up the base. The base is also representative of the industrial prevalence. Lower speed serial devices typically cost less, have fewer concerns with timing and connections. Compliance testing is necessary for high-speed serial data. Why? One of the reasons is that these standards require critical timing, encoded transmissions under exacting conditions, and higher frequencies for point-to-point transmission. It is not uncommon for a hundred tests to be performed to achieve compliance. Troubleshooting is complex and enters the realm of radio frequency for circuit board layout and connectors – what RF engineers have been challenged with for over 75 years. Case in point, if a PCI connector is not compliant it is discarded, unlike a CAN connector, typically a 9-pin sub-D connector, which can be re-soldered. For low-speed serial data timing is less critical. Transmissions are multicast, where one master device broadcasts to multiple slave devices connected by many nodes (or points) on the bus. Such troubleshooting is at the system level.

Low-Speed Serial Data Challenges: Market

- Low-cost Consumer Electronics
  - Falling prices of consumer electronics (DVD players, digital cameras, laptop PCs, etc.) with increasing levels of complexity
- Automotive Electronics
  - Simplifying automotive wiring
  - Automotive suppliers looking for feature differentiation
- Miniaturization of Designs
  - Product size is a critical feature

Consumer Electronics

- Integrated devices
- Upgradeability
- Mobility

Automotive

- High % of value in electronics
- Ecology, safety, and comfort

Communications

- Convergence of voice, video, and data
- Efficient and reliable networks

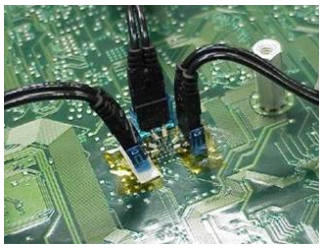
Tektronix

Much of the market drive behind the use of low-speed serial buses comes from the consumer and automotive electronics industries. For consumer electronics products, low product cost, differentiated feature capabilities, and small product size are key. As we will see, low-speed serial interconnects in embedded systems are one of the enablers for these attributes. For automotive electronics products, low system cost, lower weight, and differentiated feature capabilities are key. Again, low-speed serial interconnects are enablers. In automotive applications, the systems tend to be interconnected modules throughout the vehicle. Historically, these modules were connected with parallel cable assemblies, which were large, expensive, and very heavy. Simple electrical (or optical) serial buses simplify all of these issues.

Low-Speed Serial Data Challenges: Business

- Engineers are being tasked to do more but resources remain limited
- Making measurements needs to be easier, faster, and more accurate
- Time to market pressures, where design and debug efficiency is critical

Next Generation Signaling Needs Require New Levels of Instrumentation Performance and Analysis …

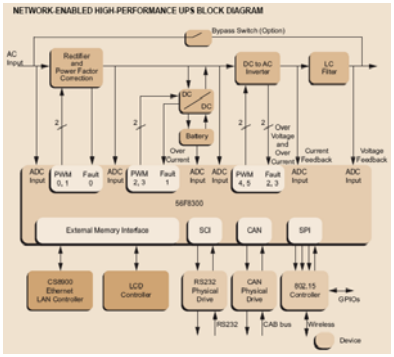


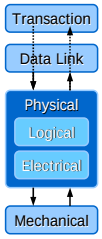
Tektronix

Research data indicates the development time has reduced from 18 months in 2000 to 13 months in 2006. Discovery of issues early in the design cycle is critical. This data supports how engineers are being asked to do more with less resources. Making measurements needs to be easy and accurate, with efficient debugging capabilities to reduce and alleviate such time-to-market pressures.

Low-Speed Serial Data Challenges: Design

- Serial communication improves circuit board designs
  - Serial interfaces are integrated into processors, ASICs, FPGAs,…
  - Fewer connections
  - Lower total component cost
- Low-speed serial standards are less demanding than high-speed serial standards
  - Timing between signals and signal integrity is less critical to bus operation
  - Compliance testing is typically not necessary
  - Bus topologies are not limited to point-to-point networks
- End-user products often contain multiple serial standards, mixed-signals, mixed-data rates, single-ended and differential signals

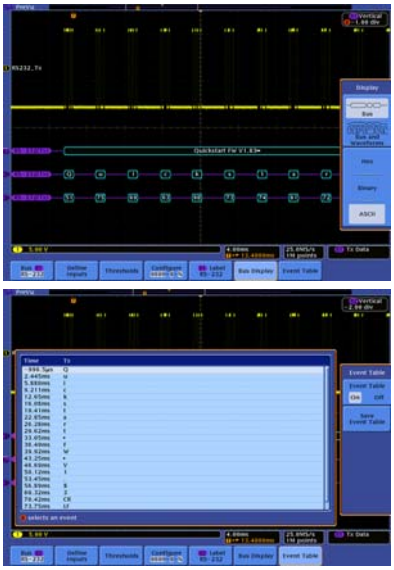




From a design perspective, serial interfaces simplify the design tasks. These interfaces are often integrated into many of the ICs already, which reduces the design and total component costs. Fewer connections also lowers the material, designs, and manufacturing costs, while increasing reliability, and improving product characteristics such as size and weight. From a technical perspective, low-speed serial buses are not particularly sensitive to timing between signals. Unlike high-speed serial buses, low-speed serial devices generally do not require compliance testing to assure proper operation. Many low-speed serial buses also allow multi-drop and multi-master bus topologies.

Low-Speed Serial Data Challenges: Decoding

- Serial bus waveforms can be manually decoded with an oscilloscope, but it is tedious and error-prone
- Hardware engineers need to verify connections and adequate signal integrity for the bus to transmit data
- Software/firmware engineers need to verify the bus messages are being sent as expected
- System engineers need to verify the system components are working together as designed
- All engineers want a simple, easy-to-use, and complete solution that is economical



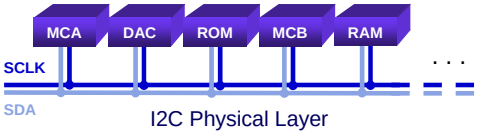
RS-232 Decoding Formats

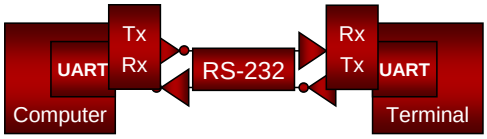
Tektronix

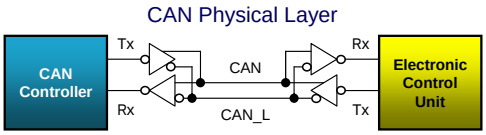
Serial signals are spread over time, making them more difficult to visualize than parallel signals. And, it depends what you're trying to visualize! Imagine a software/firmware engineer, who doesn't use an oscilloscope, but needs to verify that the software is causing a specific message to be transmitted over a serial bus. With a logic analyzer looking at the parallel bus, the engineer could see a text listing of the values and verify them against the software listing. But what do the waveforms on the display mean? Imagine a system engineer who is trying to find the root cause of a problem that causes the system to crash. Was it the software, the hardware, or some interaction between them? How does the engineer correlate the serial messages from the software to the high-level operation of the hardware circuit modules? One commonality between parallel and serial data remains true for all…all engineers want an economical solution that is simple, easy-to-use, and complete.

Low-Speed Serial Bus Review

- RS-232 (RS-422, RS-485, UART)
- I2C
- SPI
- CAN







Tektronix

For our low-speed serial bus review we will begin with a few of the most widely used low-speed serial buses in industry today, and some of their key characteristics. These serial buses are RS-232C (including RS-422, RS-485, and UART); I2C, SPI, and CAN bus.

Recommend Standard-232 Review

Start	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Opt. Data 7	Opt. Parity	Stop	Opt. Stop
1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit	1 bit

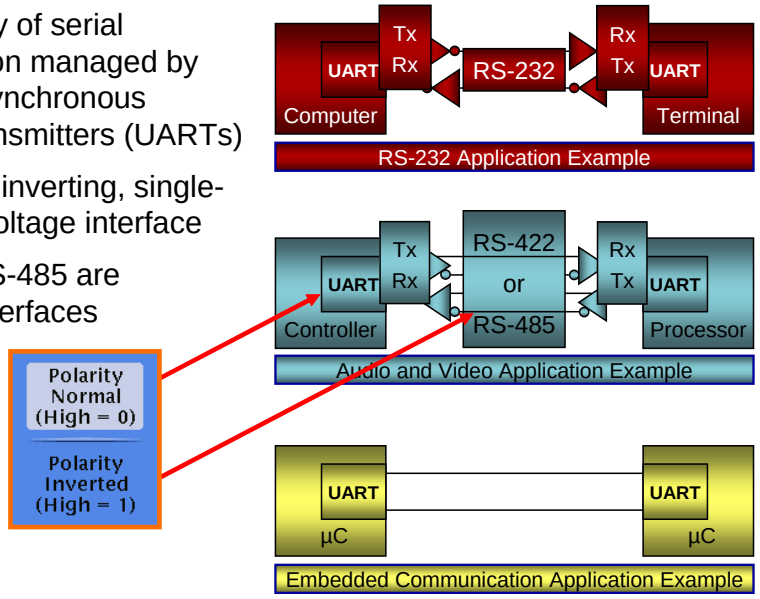
- Electronic Industries Alliance (EIA) **R**ecommended **S**tandard-232
- Two single-ended signals which provide point-to-point, full-duplex communication
- Standard does not specify character encoding, data framing, or protocols
- Since RS-232 is limited to point-to-point communication at slow speeds over fairly short distances, standards such as RS-422 and RS-485 have been developed

Tektronix

Recommended Standard-232, was developed 50 years ago in the early 1960s for interconnection between teletype terminals and modems. The standard was updated to RS-232C in 1969 to specify electrical signal characteristics, and mechanical interconnects. RS-232 uses two single-ended signals, full-duplex communication, and is limited to point-to-point communication at low speeds over fairly short distances. In the RS-232 protocol diagram we see each character begins with a Start bit, a logic “0”. The character is comprised of 7 or 8 data bits, which must also be programmed. The optional Parity bit is next. If not used, the bit is ignored. If used, the polarity must be programmed, and provides simple error detection by indicating whether there is an odd or even number of “1s” in the data word. Finally, the character is usually terminated in one or two stop bits.

RS-232 Review

- RS-232 family of serial communication managed by Universal Asynchronous Receiver/Transmitters (UARTs)
- RS-232 is an inverting, single-ended high-voltage interface
- RS-422 or RS-485 are differential interfaces

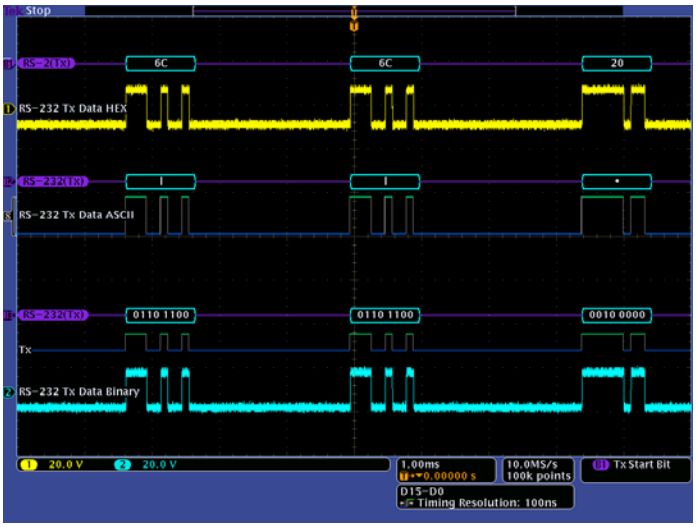


Tektronix

Since RS-232 is limited to point-to-point communication at low speeds over fairly short distances, a Universal Asynchronous Receiver/Transmitter (UART) manages all of the serial communication. These may be stand-alone ICs or a microcontroller. RS-422 specifies a balanced voltage signaling system. Data transmission is limited to a single transmitter on a pair of wires. Because of the balanced transmission and smaller voltage swings, RS-422 can transmit data at up to 10 Mbps at 1.2 meters and up to 100 kbps at 1200 meters, so it is commonly used for RS-232 extenders. RS-485 specifies the half-duplex multipoint transmission system. Like RS-422, RS-485 uses balanced transmission and can transmit data at up to 35 Mbps at 10 meters and up to 100 kbps at 1200 meters. From a debug viewpoint, both RS-422 and RS-485 use the same data format as RS-232, and the decoding process is the same. UARTs were traditionally interfaced through inverting RS-232 driver and receiver ICs. These devices use fairly high voltage (up to ±15 V) to provide simple noise immunity. However, the transmission distances are quite limited. For many embedded applications where the serial bus remains on the circuit board, designers simply connect the UARTs together (as seen in the slide). This means the signals are not inverted, as they are in RS-232. These signals are digital signals and the MSO probes are very appropriate for probing these signals.

RS-232 Solution

- Tektronix 4000 Series oscilloscope decodes the same RS-232 bus three ways:
  (1) Hex with analog signal
  (2) ASCII with digital signal
  (3) Binary with analog signal and digital waveform representation

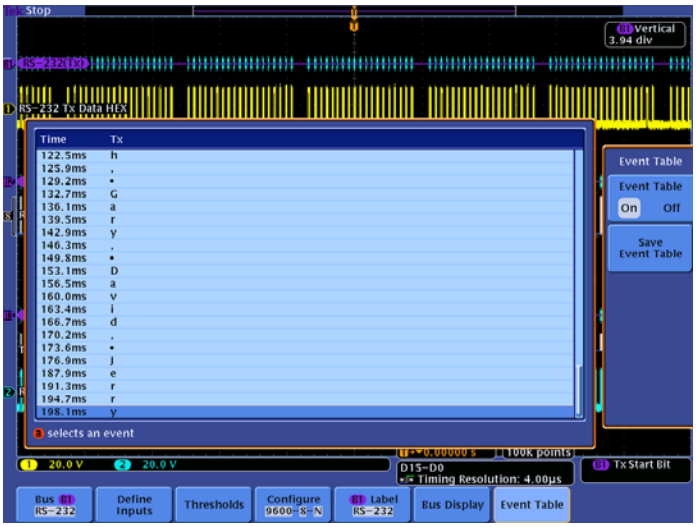


Tektronix

In the example above, the Tektronix MSO4104 mixed signal oscilloscope triggers, decodes, and analyzes the same RS-232C signal bus data in three separate ways. The first bus uses analog CH1, to verify the signal integrity of the bus, and decodes the data in hexadecimal. The next bus decodes the signal in ASCII using digital CH1. And bus three decodes the signal bus in Binary and displays the analog signal and the digital waveform representation of the signal. This technique shows all three decoding schemes correlate respective to an ASCII Table and the analog, the digital, and the digital waveform representation of the signal are validated quickly and easily.

RS-232 Solution

- Tektronix 4000 Series oscilloscope displays decoded RS-232 events (ASCII, Hex, or Binary) time correlated from trigger event

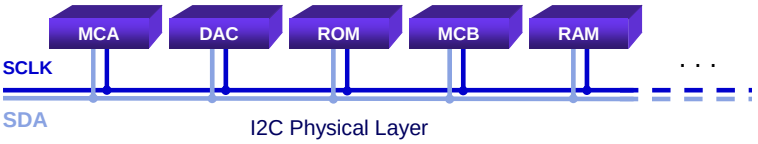


Tektronix

The Event Table displays RS-232 events from the time of the trigger position, Transmit Start, in this case, and the data decode, either in Hex, Binary, or ASCII characters or ASCII packets which combine the ASCII characters into a text looking message.

I2C (Inter-Integrated Circuit) Review

- Used for chip-to-chip communication between microcontrollers and A/Ds, D/As, FPGAs, sensors, etc.
- Uses two single-ended signals: clock and data
- Data rates: Standard Mode (100 kbps), Fast Mode (400 kbps), High Speed Mode (3.4 Mbps)
- 7-bit or 10-bit addressing modes supporting 20-30 devices can be connected to a single bus



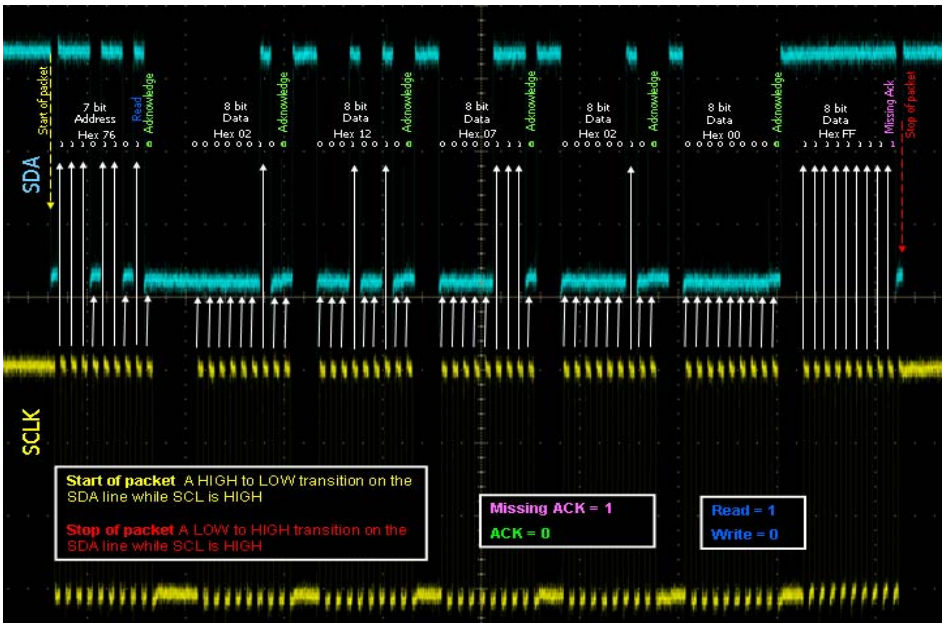
I2C Physical Layer

Start/ S_{Rep}	Addr	R/W	Ack	Data 0	Ack	Data 1	Ack	.....	Data N	Ack	Stop
1 bit	7 bits	1 bit	1 bit	8 bits	1 bit	8 bits	1 bit	0 – 8 bytes	8 bits	1 bit	1 bit

Tektronix

I2C stands for Inter-Integrated Circuit bus. The standard was developed in the 1980s by Philips to provide a low-cost way of connecting controllers to peripheral chips in TV sets. Today, it is primarily used for chip-to-chip communication. At the physical layer I2C is a 2-wire interface comprised of a bi-directional serial clock (SCL) and a data (SDA) lines. I2C supports multiple masters and slaves on the bus, but only one master may be active at one time. Initially, I2C only used 7-bit addresses, but evolved over time to allow 10-bit addressing as well. I2C supports three bit rates: 100 kbps (standard mode), 400 kbps (fast mode), and 3.4 Mbps (high-speed mode). The bits in the I2C packet shown in the diagram begin with Start – which indicates the device is taking control of the bus and a message will follow; or a Repeated Start – when a start condition occurs without a previous Stop condition. The Address is next – a 7 or 10-bit number representing the address of the device to read or write. Then the Data and then the Acknowledge from the slave device acknowledging the master's actions. The packet ends with a Stop – indicating the message is complete and the master has released the bus.

I2C Review – Hand Decoding



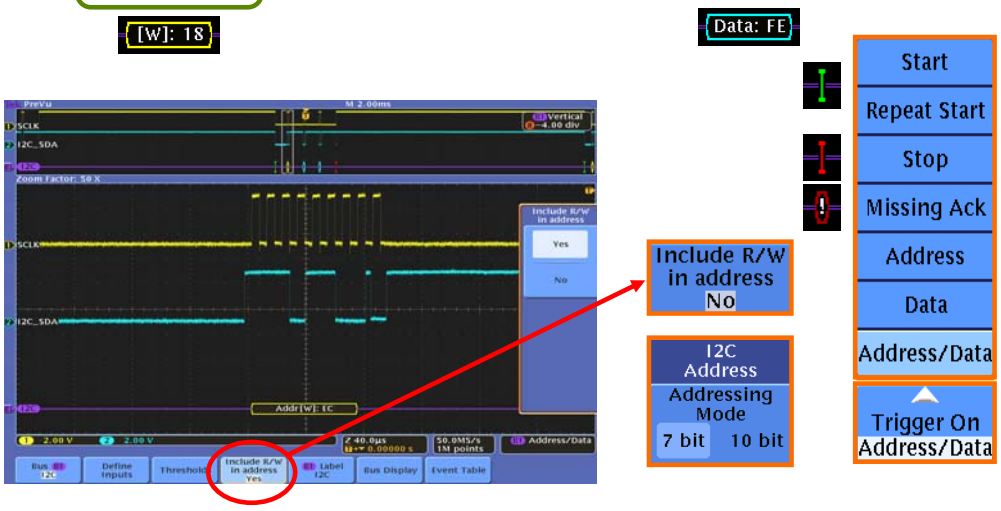
16 |

Tektronix

Even though I2C is one of the simpler serial standards to decode, hand decoding is tedious, time-consuming, and error prone. From the diagram, one can see the I2C Serial Clock and Serial Data lines. With a high-to-low transition on the data line, while the clock is high, indicates a Start of packet – when the device takes control of the bus and a message is to follow. Then decoding the Address is next, counting the next 7 consecutive edges (assuming a 7-bit address) on the clock line, while lining up the positive clock edges on the data line to determine zeros (low) and ones (high). This number represents the address of the device that will either be read from or written to. The Read/Write bit is the next bit (8th bit). 1 bit indicating a Read, 0 bit indicating a Write to the device. And, next… the phone rings and you lose count of where you were in your decoding process, and must begin again. There's got to be a better solution.

I2C Solution

Start/ S_{Rep}	Addr	R/W	Ack	Data 0	Ack	Data 1	Ack	.....	Data N	Ack	Stop
1 bit	7 bits	1 bit	1 bit	8 bits	1 bit	8 bits	1 bit	0 – 8 bytes	8 bits	1 bit	1 bit

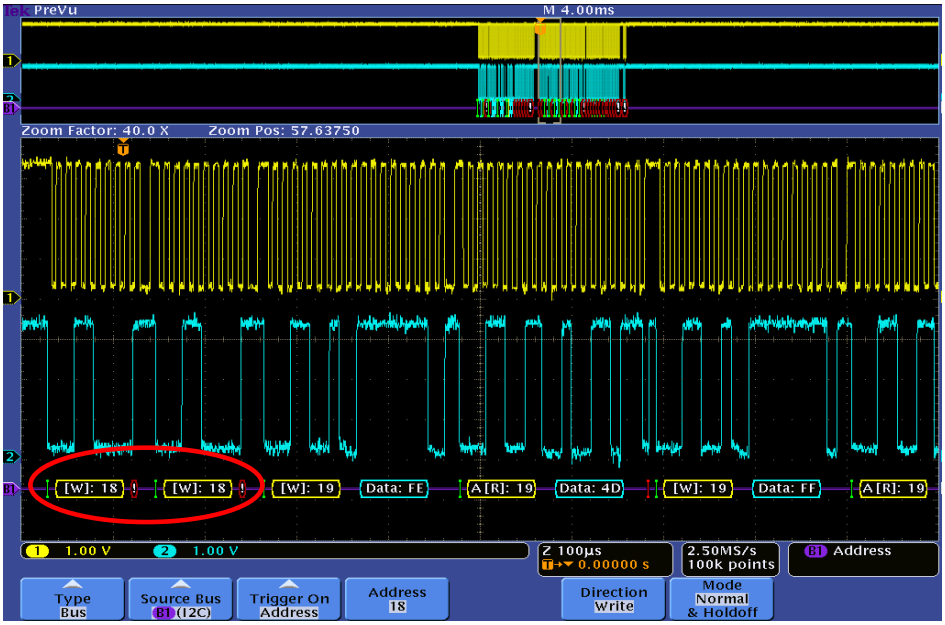


17 |

Tektronix

The MSO4000 Series can trigger and decode on every important field of the I2C protocol, including 7 and 10-bit addressing. Note how the Read/Write bits can be included in the Address, as requested by most hardware engineers. This is a much better way than to hand decode.

I2C Bus Decoding Example



I2C Bus Decoding Using Tektronix 4000 Series Oscilloscope

Tektronix

The MSO4000 Series I2C bus trigger debugs faulty temperature sensors. The engineer knows the address of the two temperature sensors, 18 and 19. The engineer easily sets up the I2C bus trigger to capture fan speed controller writing to temperature sensor 18. The fan controller attempts writing to temperature sensor at address 18 but receives a No Ack, then tries again and receives a No Ack again, as seen in the example. The fan controller then writes to the second temperature senor at address 19; the Write is successful. An Ack and then a Repeated Start, followed by a Read with the temperature information, completes the packet transfer.

SPI (System Peripheral Interface) Review

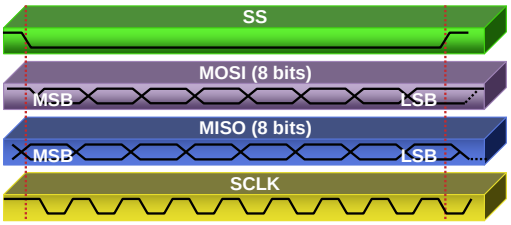
- Network can use 2-, 3-, or 4-wire bus topology
- A serial synchronous, full-duplex, multi-master, layered communication network
- Data rates up to 10 Mbps, are simultaneously transmitted and received
- Multiple bus configurations are allowed
- Used primarily to communicate between microcontrollers and their immediate peripheral devices
- Developed in the 1980s by Motorola to connect peripheral ICs to the 68000 microprocessor

SS (Slave Select) – enables slave device to accept data

MOSI – data from the master to a slave

MISO – data from a slave to the master

SCLK – serial clock

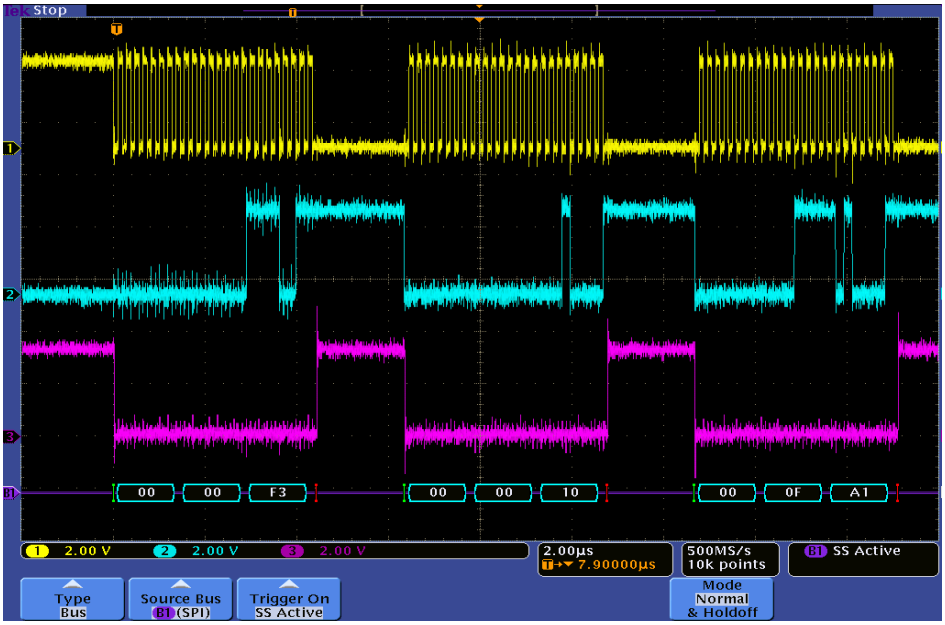


19 |

Tektronix

The System Peripheral Interface (SPI) standard is over 20 years old, created by Motorola, in the 1980s, in an effort to connect peripheral ICs to the 68000 microprocessor. SPI is a master/slave, 2 to 4-wire serial bus used primarily to communicate between the microcontroller and its systems. Whenever two devices communicate on the bus, one is referred to as the master and the other the slave. The master drives the serial clock. Data is simultaneously transmitted and received. SPI uses the Slave Select line to specify which device is being transferred to or from, at bitrates up to 10 Mbps. As such, each unique device on the bus needs its own Slave Select signal from the master.

SPI Bus Decoding Example



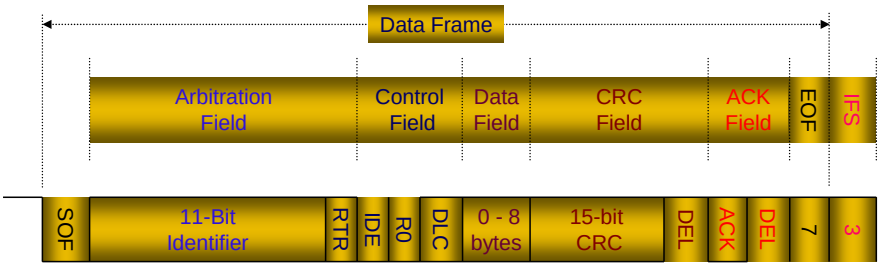
SPI Bus Decoding Using Tektronix 4000 Series Oscilloscope

20 |

Tektronix

The DPO4EMBD (Embedded Serial Triggering and Analysis Module) adds optional SPI serial triggering, decoding, and searching to the MSO4000 and DPO4000 products. This support presents the features in the language of the standard and allows the customer to examine all of the critical elements of the serial signal.

CAN (Controller Area Network) Review



- A serial asynchronous, multi-master, layered communication network
- Used for system-to-system communication in Automotive, Industrial Automation, and Medical Equipment
- Data rates from 10 kbps to 1 Mbps
- Sophisticated error detection and error handling mechanisms
- Flexible signaling support for low-cost implementation
- Physical bus is single-wire, dual-wire, and fault tolerant

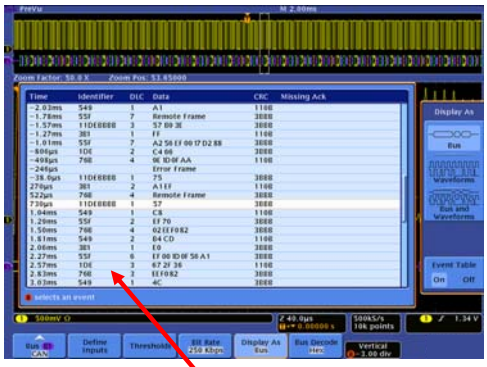
21 |

Tektronix

The Controller Area Network (CAN) bus is another serial standard that has been used in industry for over 20 years. Initially, CAN was used in the automotive industry to reduced the number of wires and connections. For example, some cars use as much as 5 kilometers of wire. Mercedes-Benz was the first automobile manufacturer to employ CAN bus, in 1980, in certain models. Improving electrical noise immunity in automotive electronics is desired. CAN provides a balanced differential, 2-wire interface, which uses Shielded Twisted Pair, Un-shielded Twisted Pair, or a ribbon cable. CAN supports bitrates up 10 kbps to 1 Mbps, and uses NRZ encoding and bit stuffing to keep the signal balanced. CAN is used in many different industries today. In the diagram above we see the most common of the four CAN frames; the Data frame and the fields of which it is composed being the Start of Frame, and the 11-bit ID Frame. CAN 2.0 supports Extended Addressing (29 bits). CAN protocol is non-trivial to hand decode. Using some kind of decoding and analysis scheme is almost a necessity. Let's look at one solution.

CAN Solution

Search and Mark Feature



Event Table Displays Decoded
CAN Message Frames with Timestamps

Automotive

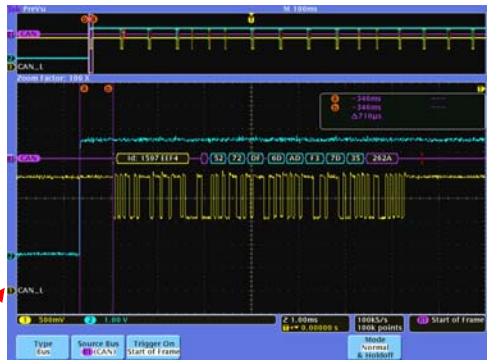
- LIN and FlexRay are also supported with similar capabilities

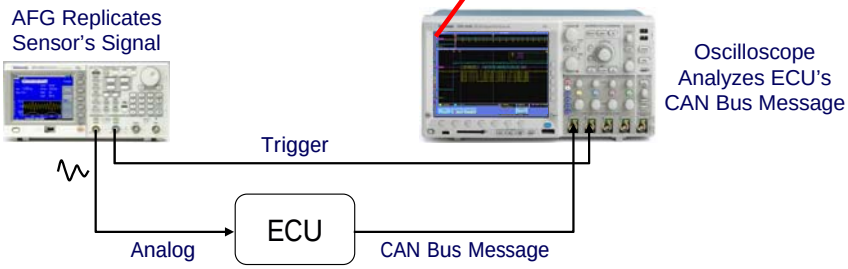
Tektronix

The Tektronix MSO/DPO4000 Series can trigger, decode, and analyze all four CAN Frames. A Search and Mark feature, shown on the left, allows the user to make use of 10 Mpoint record length data. The data can be analyzed quickly by allowing the oscilloscope to search specific CAN fields of interest and then mark the found events for deep analysis later. The Event Table (right side) displays each frame in the record by position from trigger, and decodes the frame identifier, data length character, data, CRC, and missing acknowledgements. The powerful functionality is similar to that of a network analyzer, but with the oscilloscope you also have the analog signal to view.

Solution Example: AFG3252 & MSO4104

- Evaluation of CAN Bus ECU Performance
  - Oscilloscope validates ECU's CAN bus message
  - Oscilloscopes measures latency of sensor/ECU chain





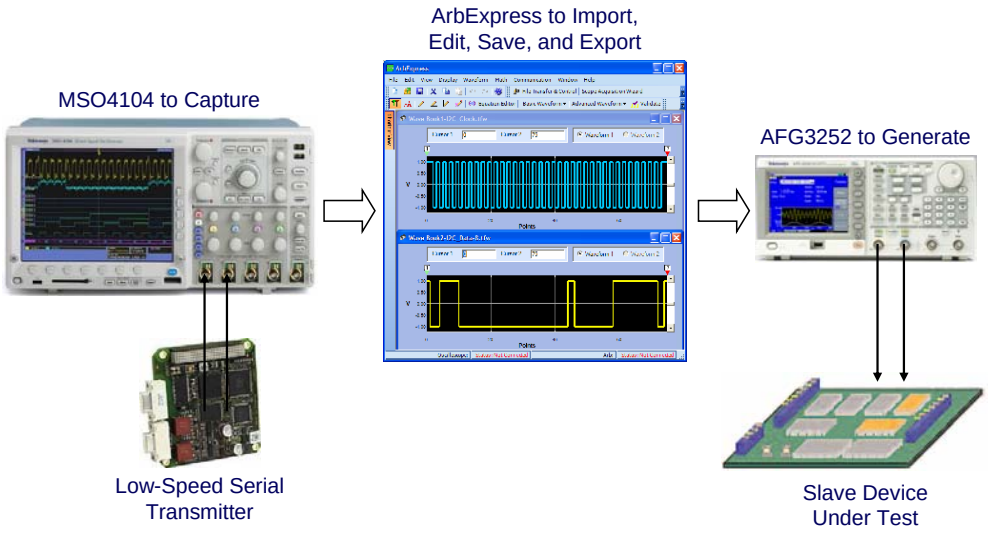
23 |

Tektronix

Within a CAN bus network, Electronic Control Units (ECUs) are deployed to communicate with devices such as actuators and sensors with analog inputs and outputs. For the designer of the ECU, it is important to validate the ECU's CAN bus message in response to an analog stimulus, and to measure the time latency of sensors, i.e. the time it takes from the analog stimulus until the ECU sends the appropriate CAN bus message. The block diagram here illustrates the measurement setup. An arbitrary/function generator (AFG) stimulates the ECU with a standard function or custom waveform, and an oscilloscope captures and analyzes the ECU CAN bus message. Via a trigger line from the AFG that runs synchronously to the analog stimulus, the oscilloscope can determine the sensor latency. An example where such measurements can be relevant is steering wheel angle sensors. In this case, the AFG would replicate the angle sensor signal, and the oscilloscope tracks the delay of the ECU message versus steering wheel angle changes.

Tektronix Measurement Solution

Creating Low-Speed Serial Signals



24 |

Tektronix

When designing and testing low-speed serial devices, a mechanism is required to replace transmit signals from missing components. In many cases it may also be useful to add noise or other anomalies to the signals to test the devices under real-world and stress conditions. A common approach is to create low-speed serial signals using software applications or to simply capture a live signal using an oscilloscope. The created or captured waveform can then be imported into the ArbExpress software for editing and transfer to an Arbitrary/Function Generator (AFG). The AFG can replicate the signal repeatedly to test the final circuit design in a controlled environment such as a temperature chamber or EMC testing room. Since replicated signals can be easily modified they allow control over the testing to verify full functionality and reliability of the device.

Complete Solution for Low-Speed Serial Data

Tektronix Oscilloscopes

- MSO/DPO4000 Series Oscilloscopes
  - MSO/DPO4032, MSO/DPO4034, MSO/DPO4054, MSO/DPO4104
  - Serial Triggering and Decoding with Application Modules
  - Application Modules
    - DPO4AUTO: Automotive Serial Triggering and Analysis Module (CAN, LIN)
    - DPO4EMBD: Embedded Serial Triggering and Analysis Module (I2C, SPI)
    - DPO4COMP: Embedded Serial Triggering and Analysis Module (RS-232, RS-422, RS-485, UART)
    - DPO4AUTOMAX: Extended Automotive Serial Triggering and Analysis Module (CAN, LIN, FlexRay)
- DPO7000 Series Oscilloscopes
  - DPO7054, DPO7104, DPO7254
  - I2C, SPI, and RS-232 Triggering is Standard
  - Application Modules
    - TDSVNM: Serial Trigger, Decode, and Analysis for CAN and LIN
    - Option LSA used with ATM-1 (Automotive Trigger Module)
      - Dedicated Trigger Module for CAN for Logical and Advanced Triggering
    - Additional oscilloscope-resident application support provided by Prodigy Test Solutions
      - PDI-R I2C Protocol Decode Software
      - PDS-R SPI Protocol decode software
      - PDF-R FlexRay Protocol decode software

Tektronix

When we look at what is needed for a complete solution to solve our serial data triggering, decoding, and analysis needs we begin with a measurement instrument such as the MSO/DPO4000 Series which supports serial triggering, and applications for RS-232C/RS-422/RS-485 and UARTs, I2C, SPI, CAN, LIN, and FlexRay. Also the DPO7000 Series offers I2C, SPI, and RS-232C triggering (standard), and integrated third-party hardware using the Automotive Trigger Module (ATM-1). Application modules are also available like TDSVNM CAN/LIN and Prodigy Test Solutions software which supports decoding and analyzing I2C, SPI, and FlexRay.

Complete Solution for Low-Speed Serial Data

Tektronix Probes and Signal Sources

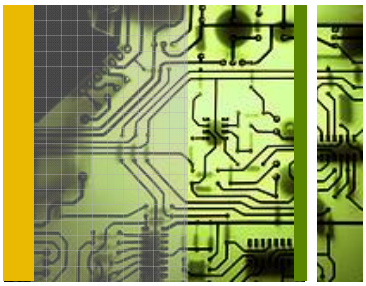
- Differential and Single-Ended TekVPI Probes
  - TDP1000, TDP0500, and TCP0030 are well suited for CAN bus signals
  - All TekVPI Probes Provide:
    - Connect directly to the MSO/DPO4000 Series oscilloscopes
    - TekVPI probes are automatically deskewed to within 1 ns
    - Automatic units scaling and readout on the oscilloscope display
    - Built-in probe calibration routines
    - Probe controlled via switches on the probe or through the oscilloscope probe menu
- Tektronix Signal Sources
  - AFG3000 Series
    - Simulate I2C, CAN, LIN, RS-232 data and clock signals
    - Generate analog sensor signals for ECU evaluation
    - Up to 2 analog or digital outputs
  - AWG5000 Series
    - Generate analog, digital and mixed signals
    - Up to 4 analog outputs, 28 digital outputs, and 8 variable level marker outputs
    - Real-time waveform sequencing for loops, jumps, patterns and conditional branches

Tektronix

The solution is completed by offering Differential, Single-Ended, and Current probes; Signal Sources, like the AFG3200 Series to stimulate Electronic Control Units, and the advanced AWG5000 Series, with 28 digital outputs, to solve your serial data triggering, decoding, and analysis needs.

Summary

- Low-Speed Serial presents a unique set of measurement and analysis needs
- Making measurements needs to be easier, faster, and more accurate
- Solutions are needed at the physical and data-links layers, for triggering, decoding, and analyzing low-speed serial networks
- All engineers want a simple, easy-to-use, and complete solution that is economical

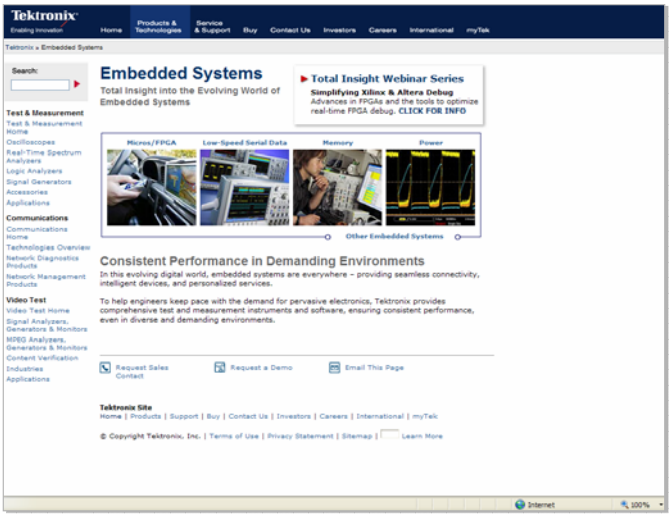


Tektronix

Low-speed serial data presents a unique set of measurement and analysis needs and challenges. Making such measurements needs to be easier, faster, and more accurate with solutions for both the physical and protocol layers, for triggering, decoding, and analyzing the low-speed serial data networks that have changed or transitioned from parallel to serial data. What has not changed however is what all engineer's want…an economical, simple, easy-to-use, and complete solution.

Additional Information

- Tektronix
  - www.tektronix.com/embedded



For additional information and the latest details, please go to www.tektronix.com/embedded.

Thank You!

Tektronix

Thank you for joining us on this session, “Efficiently Triggering, Debugging, and Decoding Low-Speed Serial Buses”.