probe-rs & FTDI Probes

•••

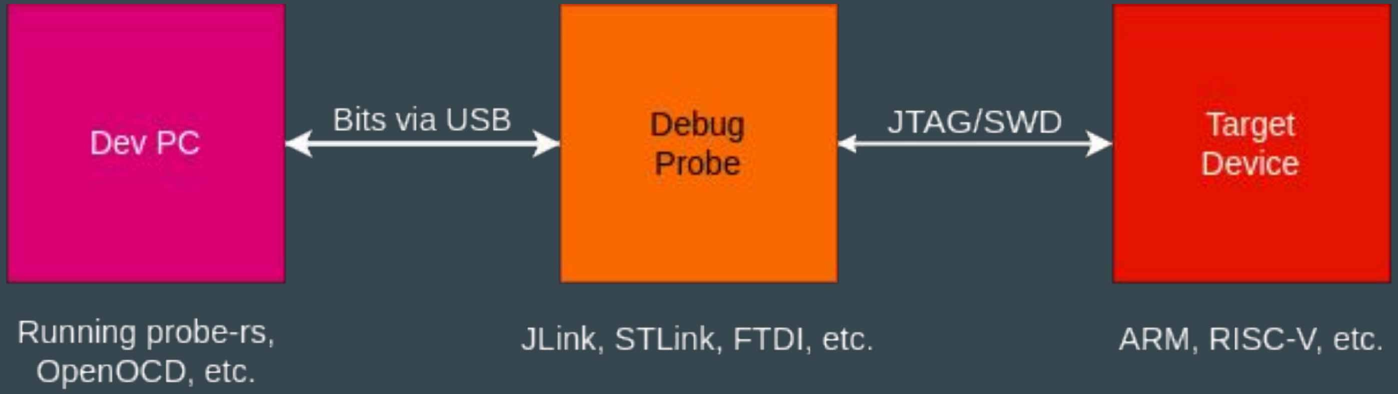
Alex Gavin

github.com/a-gavin

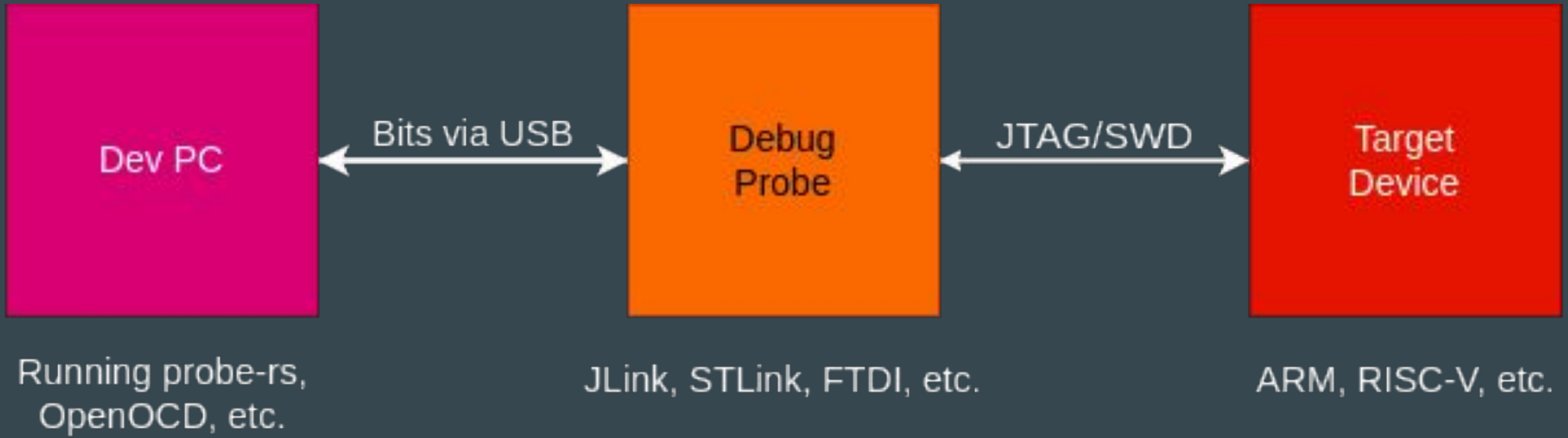
PSA: Not affiliated w/ FTDI or probe-rs

probe-rs Overview

- "A modern, embedded debugging toolkit, written in Rust"
- Somewhat similar to OpenOCD
  - Tells debug probes to tell target devices what to do
- Support for ARM and RISC-V



Embedded Debugging Primer



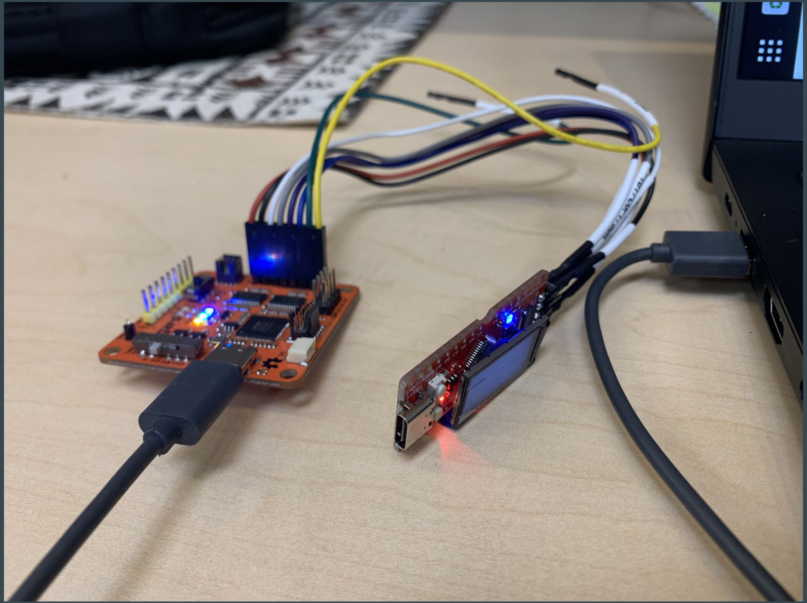
probe-rs Overview

- Integrations for flexible use:
    - GDB
    - VSCode
    - Cargo tools `cargo embed` and `cargo flash`
- API for programmable target debugging

  https://docs.rs/probe-rs/latest/probe_rs/
- Supported debug probes:
    - J-Link, ST-Link, CMSIS-DAP, FTDI, ESP USB JTAG

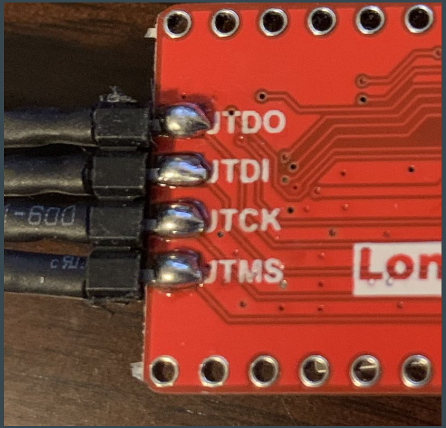
My Debug Setup

- Devices:
  - Dev PC: Linux machine
  - Debug Probe: 1BitSquared Tigard
  - Target Device: Longan Nano RISC-V


Probe and Target

Connecting Probe to Target

- Hook up VCC (power) and GND
- Hook up JTAG pins:
  - TMS - Test Mode Select
  - TDI - Test Data In (to target)
  - TDO - Test Data Out (from target)
  - TCK - Test Clock
  - TRST (optional) - Test Reset

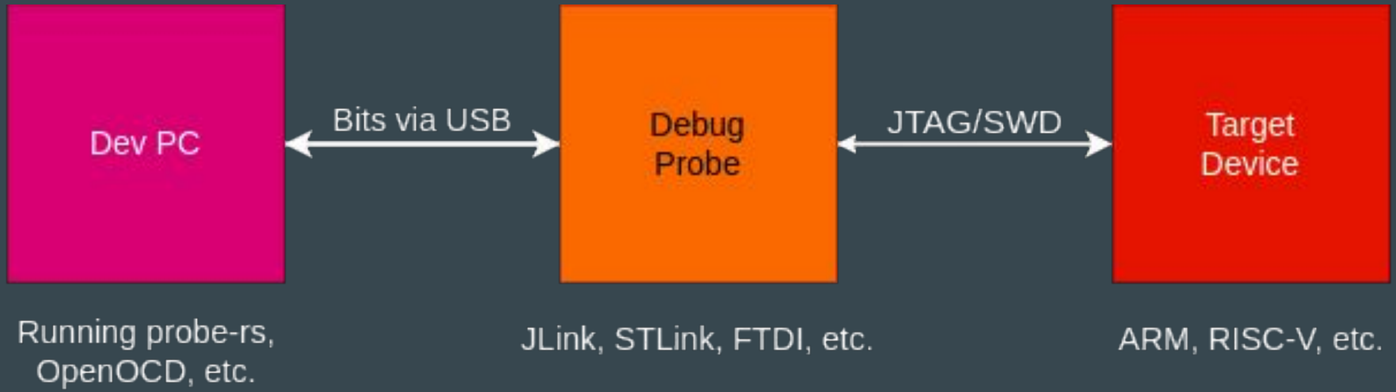


Target JTAG Pins

Target: Device debug probe attached to; the device you're programming

probe-rs and Debugging

- Implements JTAG and SWD host-side
- Debug probe is medium of communication to target
- “Bits via USB” different for different debug probes
  - CMSIS-DAP commands via USB HID
  - FTDI, JLink, etc: Bit bashing



FTDI Bit-Bashing

- FTDI MPSSE
  - USB to synchronous protocol (e.g. serial, JTAG, SWD)
  - Series of commands to transfer data

FTDI Bit-Bashing

- FTDI MPSSE
  - USB to synchronous protocol (e.g. serial, JTAG, SWD) ← Can be any, programmer gets to define
  - Series of commands to transfer data

FTDI Bit-Bashing

- FTDI MPSSE
  - USB to synchronous protocol (e.g. serial, JTAG, SWD) ← Can be any, programmer gets to define
  - Series of commands to transfer data
- Example: Write data to chip memory*
  - Get data from higher layers of probe-rs
  - Direct target to required state for write to memory
  - Package data up into FTDI MPSSE commands
  - Write data to FTDI probe

* Doing some hand-waving here

Example MPSSE Commands (All Sent out via USB)

Clock Data Bytes In and Out (LSB first)

39	06	00	FF	FF	FF	FF	FF	FF	FF
Cmd	Length (bytes)		Data (to TDI)						

Clock Data to TMS Pin (LSB first, No Read)

4B	04	03
Cmd	Length (bits)	Data (to TMS)

Disable Loopback

85
Cmd

Key:

- 🟥 Command
- 🟨 Length
- 🟩 Data

All values are hex

Example MPSSE Commands

“All data sent out via USB”:
A write of 0x4B0403 to an open USB file descriptor will perform the “Clock Data to TMS Pin” command on bottom of slide
fd.write([0x4B, 0x04, 0x03] as [u8])?; ← Rust pseudocode

FTDI chip unpacks data and drives desired pins

Clock Data Bytes In and Out (LSB first)

39	06	00	FF	FF	FF	FF	FF	FF	FF
Cmd	Length (bytes)		Data (to TDI)						

Clock Data to TMS Pin (LSB first, No Read)

4B	04	03
Cmd	Length (bits)	Data (to TMS)

Disable Loopback

85
Cmd

Key:

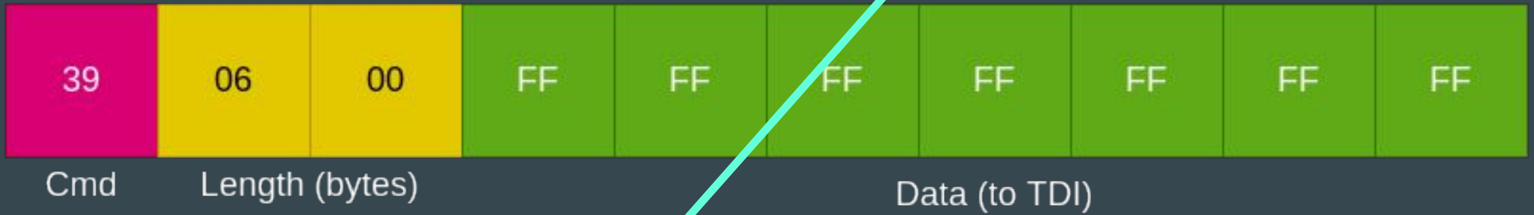
🟥	Command
🟨	Length
🟩	Data

All values are hex

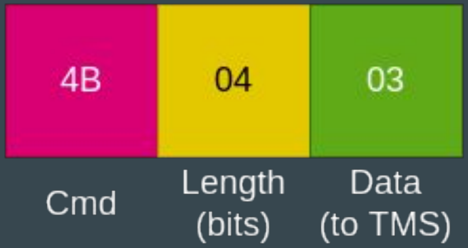
Example MPSSE Commands

Data to transfer to target "packaged up"

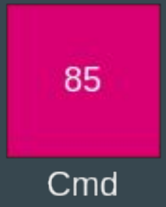
Clock Data Bytes In and Out (LSB first)



Clock Data to TMS Pin (LSB first, No Read)



Disable Loopback



Key:

- 🟥 Command
- 🟨 Length
- 🟩 Data

All values are hex

probe-rs FTDI Debug Probe

- Currently only supports JTAG for RISC-V (as of v0.15.0)
- Other probe-rs debug probes support SWD for ARM
  - JLink supports both
- My work is focused on:
  - Refactoring FTDI JTAG implementation (currently)
  - Implementing SWD for ARM for FTDI probes (up next!)

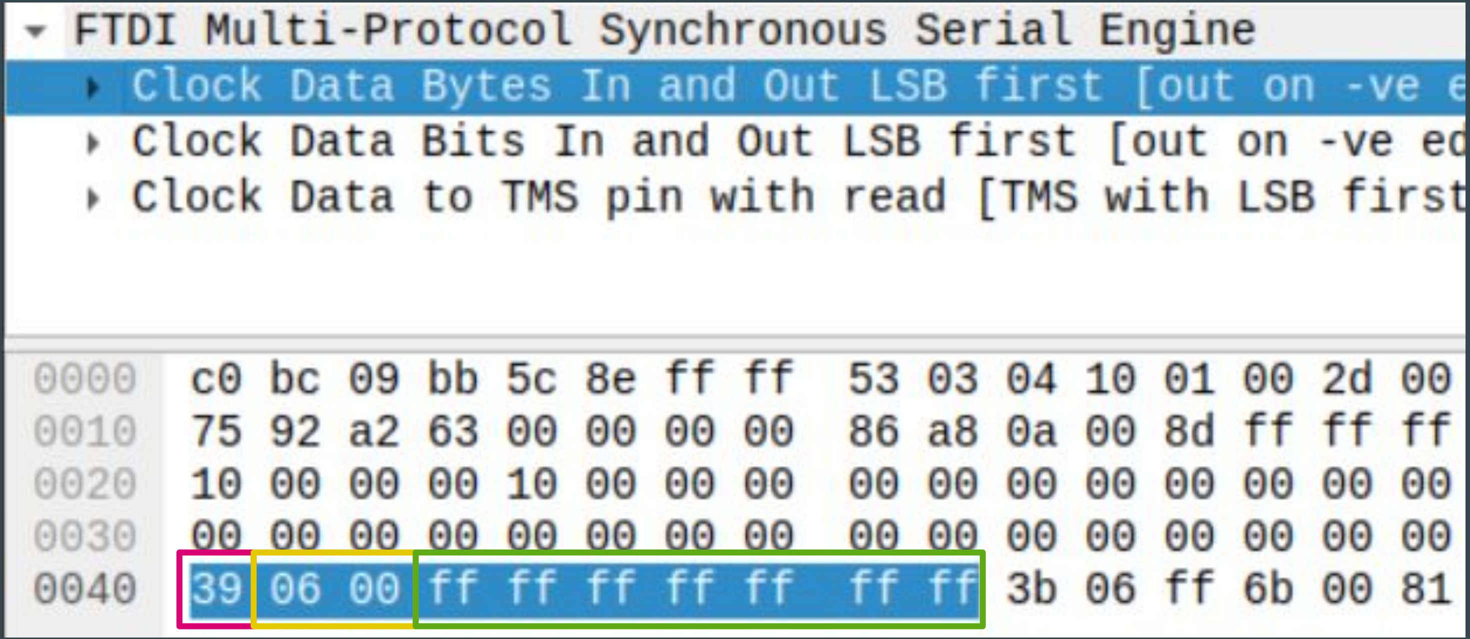
Appendix A: Wireshark for Debugging probe-rs

- Great when coupled with signal analyzer (e.g. BitMagic Basic)
  - Needed because Wireshark good but can't do everything (e.g. visualize/compare transfer timings)
- Setup here: https://wiki.wireshark.org/CaptureSetup/USB

RE: BitMagic Basic

Not affiliated, just rly like it! Can use with Sigrok Pulseview software (both are open source!)

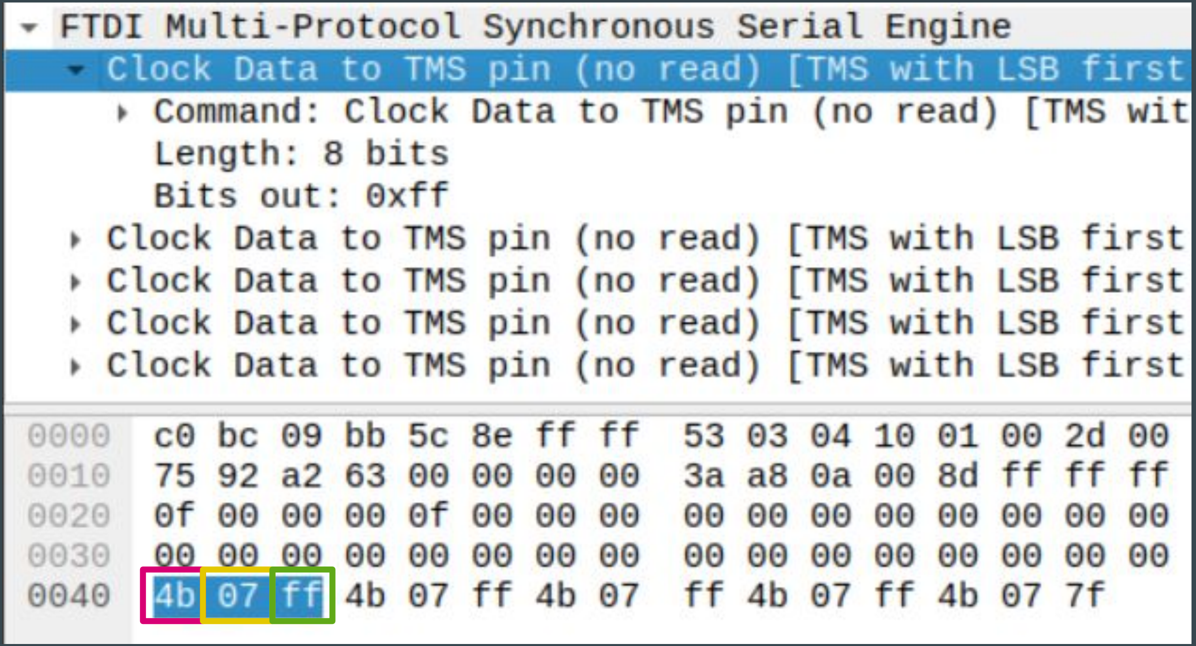
Wireshark Capture Example



```
▾ FTDI Multi-Protocol Synchronous Serial Engine
   ▸ Clock Data Bytes In and Out LSB first [out on -ve e
   ▸ Clock Data Bits In and Out LSB first [out on -ve ed
   ▸ Clock Data to TMS pin with read [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  86 a8 0a 00 8d ff ff ff
0020  10 00 00 00 10 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  39 06 00 ff ff ff ff ff  ff ff 3b 06 ff 6b 00 81
```

Transfer six bytes to and from the target (LSB first)

Using Wireshark to Debug probe-rs

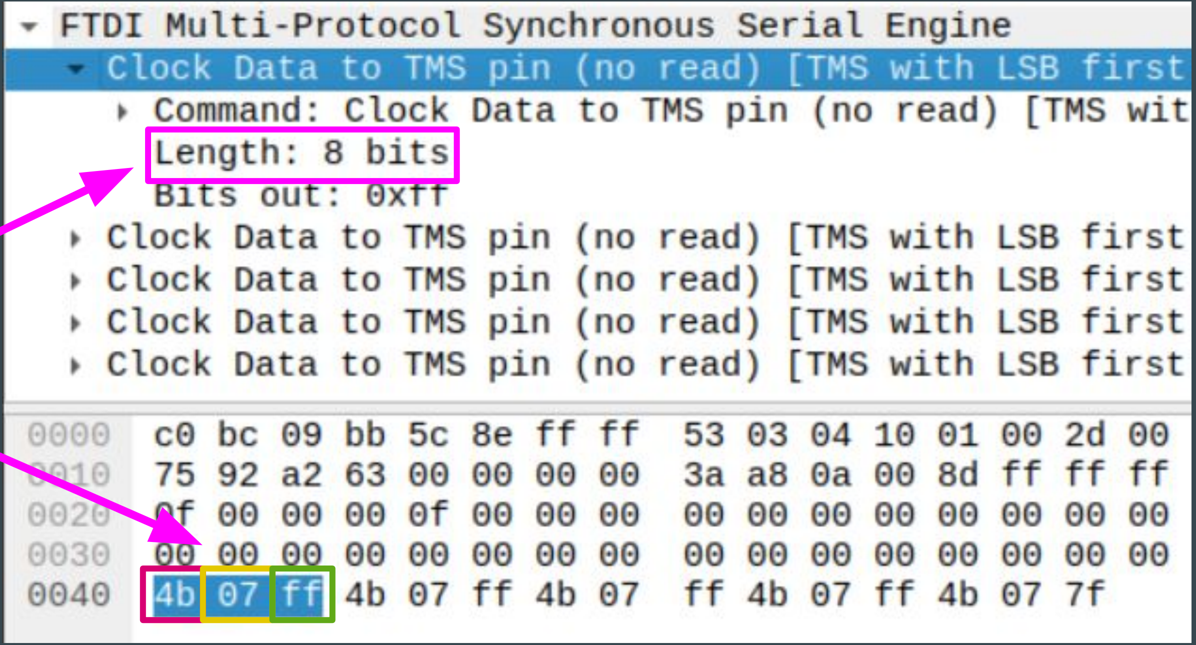


```
▾ FTDI Multi-Protocol Synchronous Serial Engine
  ▾ Clock Data to TMS pin (no read) [TMS with LSB first
    ▸ Command: Clock Data to TMS pin (no read) [TMS wit
      Length: 8 bits
      Bits out: 0xff
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  3a a8 0a 00 8d ff ff ff
0020  0f 00 00 00 0f 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  4b 07 ff 4b 07 ff 4b 07  ff 4b 07 ff 4b 07 7f
```

Clock data to TMS (LSB first, no read)

Using Wireshark to Debug probe-rs

Little lie!



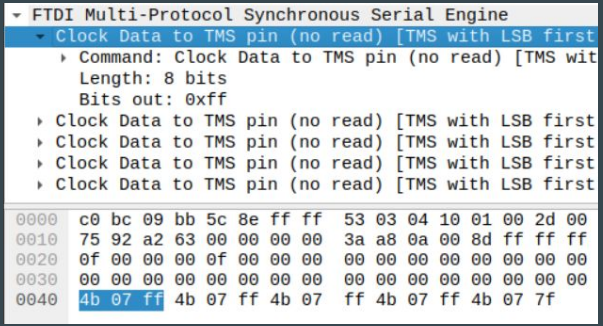
```
▾ FTDI Multi-Protocol Synchronous Serial Engine
  ▾ Clock Data to TMS pin (no read) [TMS with LSB first
    ▸ Command: Clock Data to TMS pin (no read) [TMS wit
      Length: 8 bits
      Bits out: 0xff
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  3a a8 0a 00 8d ff ff ff
0020  0f 00 00 00 0f 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  4b 07 ff 4b 07 ff 4b 07  ff 4b 07 ff 4b 07 7f
```

Clock data to TMS (LSB first, no read)

...datasheet says max length of seven bits

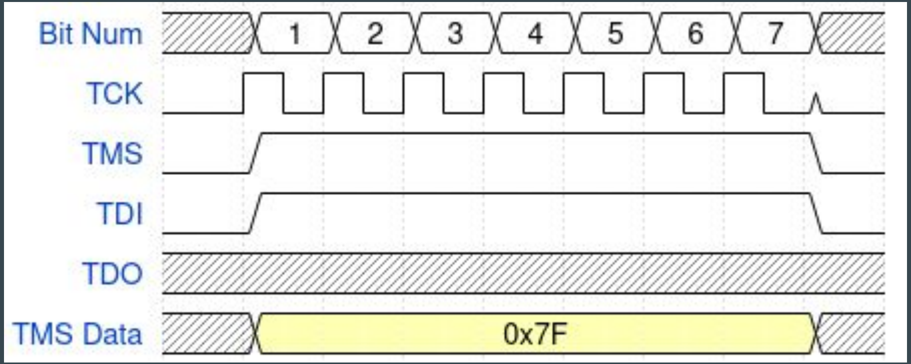
(length field is 0-indexed, so 0x07 is len 8)

Using Wireshark to Debug probe-rs

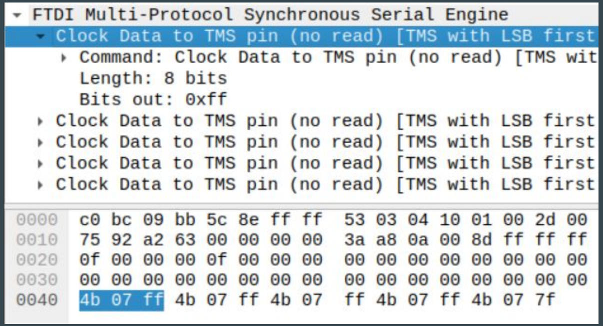


```
▾ FTDI Multi-Protocol Synchronous Serial Engine
  ▾ Clock Data to TMS pin (no read) [TMS with LSB first
    ▸ Command: Clock Data to TMS pin (no read) [TMS wit
      Length: 8 bits
      Bits out: 0xff
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  3a a8 0a 00 8d ff ff ff
0020  0f 00 00 00 0f 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  4b 07 ff 4b 07 ff 4b 07  ff 4b 07 ff 4b 07 7f
```

What transfer looks like on the wire

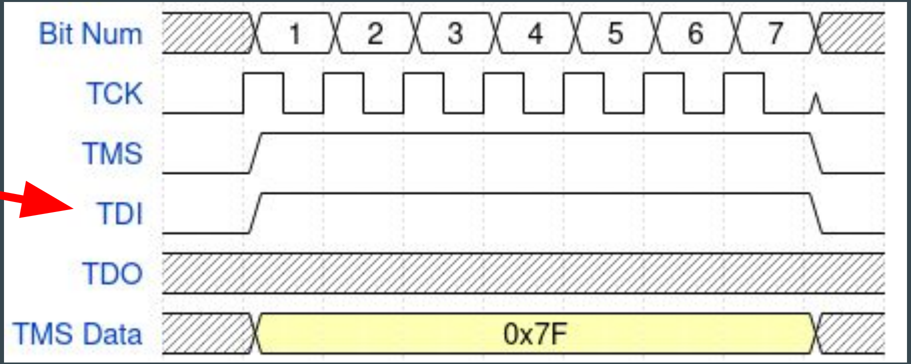


Using Wireshark to Debug probe-rs



```
▾ FTDI Multi-Protocol Synchronous Serial Engine
  ▾ Clock Data to TMS pin (no read) [TMS with LSB first
    ▸ Command: Clock Data to TMS pin (no read) [TMS wit
      Length: 8 bits
      Bits out: 0xff
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  3a a8 0a 00 8d ff ff ff
0020  0f 00 00 00 0f 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  4b 07 ff 4b 07 ff 4b 07  ff 4b 07 ff 4b 07 7f
```

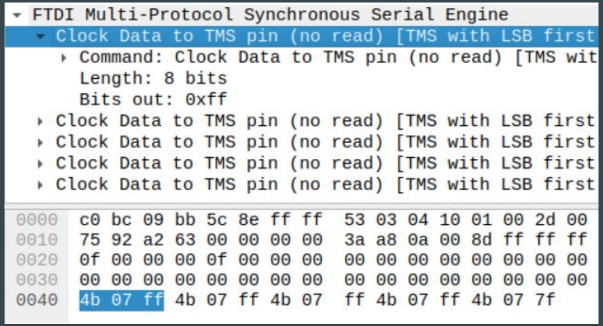
What transfer looks like on the wire



Huh, that's weird.... TDI is driven high for the duration of the transfer

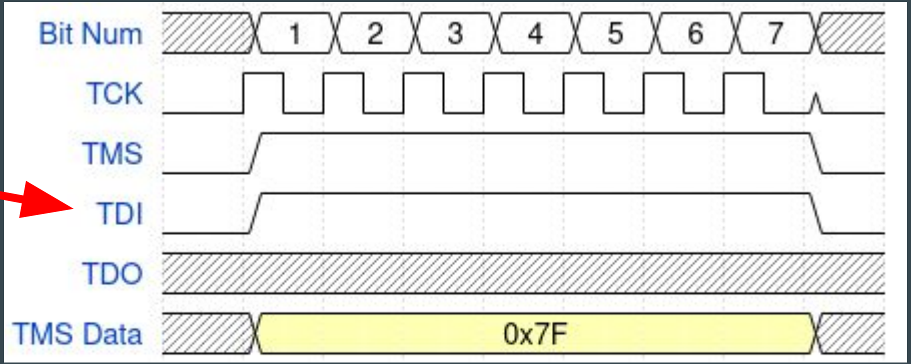
We didn't tell it to do that?

Using Wireshark to Debug probe-rs



```
▾ FTDI Multi-Protocol Synchronous Serial Engine
  ▾ Clock Data to TMS pin (no read) [TMS with LSB first
    ▸ Command: Clock Data to TMS pin (no read) [TMS wit
      Length: 8 bits
      Bits out: 0xff
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first
  ▸ Clock Data to TMS pin (no read) [TMS with LSB first

0000  c0 bc 09 bb 5c 8e ff ff  53 03 04 10 01 00 2d 00
0010  75 92 a2 63 00 00 00 00  3a a8 0a 00 8d ff ff ff
0020  0f 00 00 00 0f 00 00 00  00 00 00 00 00 00 00 00
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0040  4b 07 ff 4b 07 ff 4b 07  ff 4b 07 ff 4b 07 7f
```

What transfer looks like on the wire

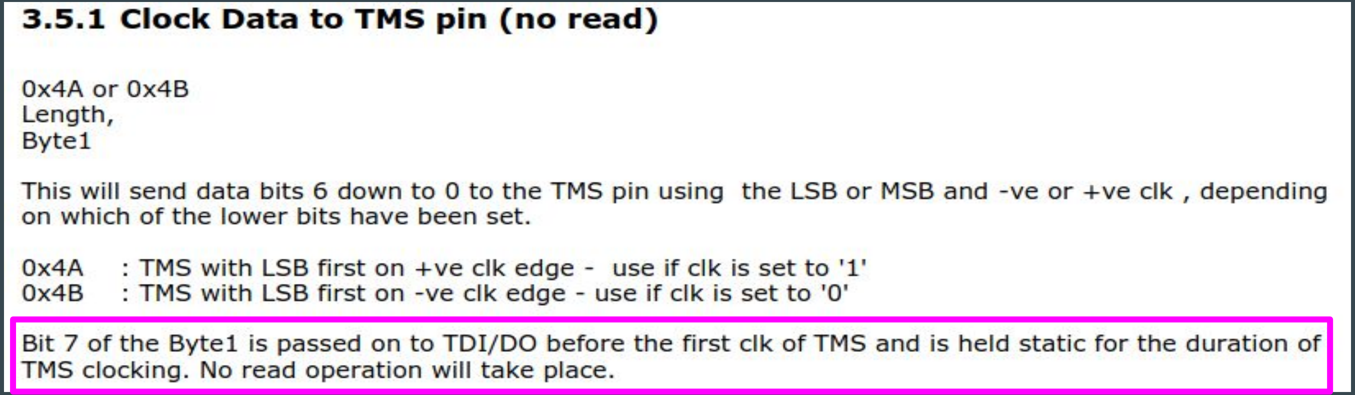


Huh, that's weird.... TDI is driven high for the duration of the transfer

We didn't tell it to do that?

Let's check the datasheet

Using Wireshark to Debug probe-rs



3.5.1 Clock Data to TMS pin (no read)

0x4A or 0x4B
Length,
Byte1

This will send data bits 6 down to 0 to the TMS pin using the LSB or MSB and -ve or +ve clk , depending on which of the lower bits have been set.

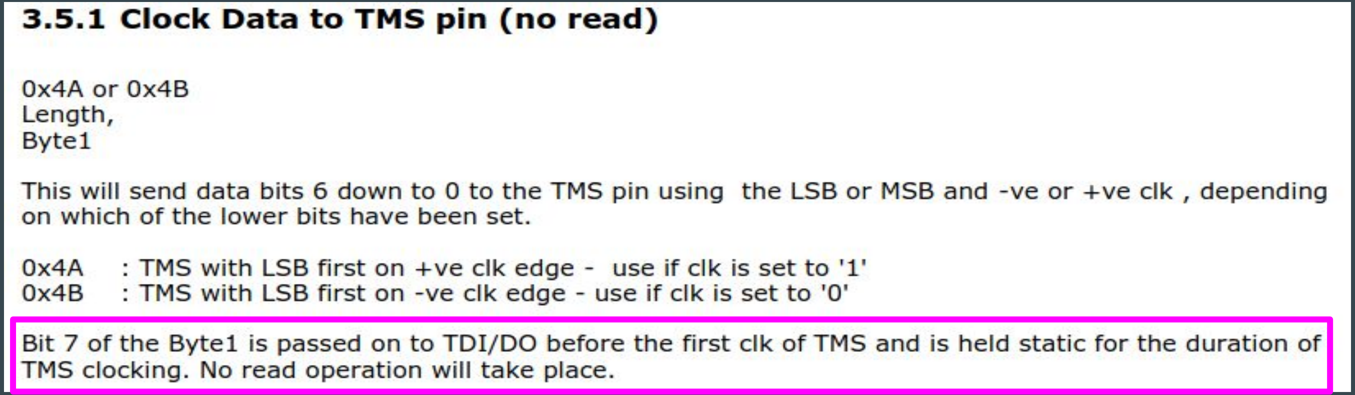
0x4A	: TMS with LSB first on +ve clk edge - use if clk is set to '1'
0x4B	: TMS with LSB first on -ve clk edge - use if clk is set to '0'

Bit 7 of the Byte1 is passed on to TDI/DO before the first clk of TMS and is held static for the duration of TMS clocking. No read operation will take place.

Eighth bit goes to TDI......**always**

Screenshot from FTDI MPSSE datasheet

Using Wireshark to Debug probe-rs



3.5.1 Clock Data to TMS pin (no read)

0x4A or 0x4B
Length,
Byte1

This will send data bits 6 down to 0 to the TMS pin using the LSB or MSB and -ve or +ve clk , depending on which of the lower bits have been set.

0x4A : TMS with LSB first on +ve clk edge - use if clk is set to '1'
0x4B : TMS with LSB first on -ve clk edge - use if clk is set to '0'

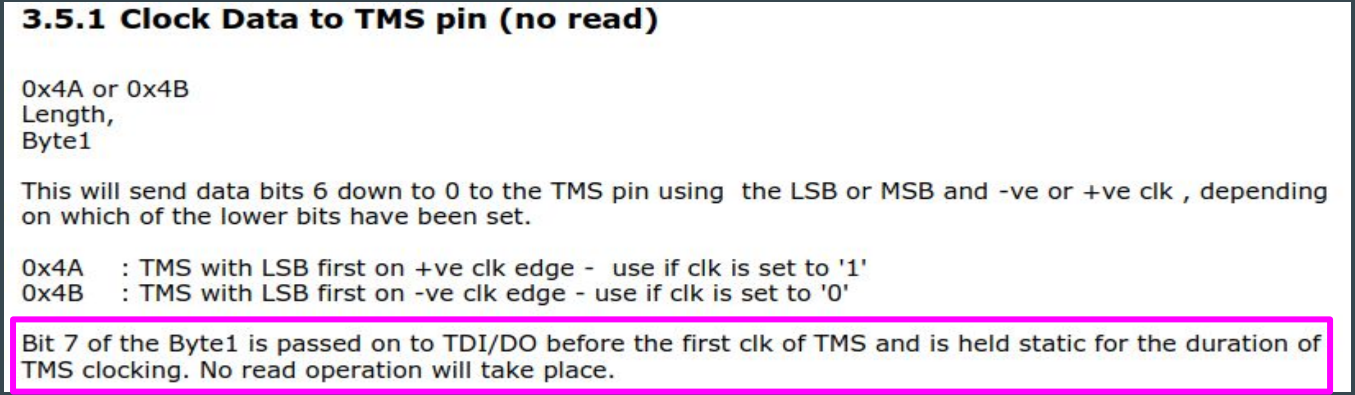
Bit 7 of the Byte1 is passed on to TDI/DO before the first clk of TMS and is held static for the duration of TMS clocking. No read operation will take place.

Eighth bit goes to TDI......**always**

Takeaway: *Always* put last bit clocked to TDI in next TMS txfr

Screenshot from FTDI MPSSE datasheet

Using Wireshark to Debug probe-rs



3.5.1 Clock Data to TMS pin (no read)

0x4A or 0x4B
Length,
Byte1

This will send data bits 6 down to 0 to the TMS pin using the LSB or MSB and -ve or +ve clk , depending on which of the lower bits have been set.

0x4A : TMS with LSB first on +ve clk edge - use if clk is set to '1'
0x4B : TMS with LSB first on -ve clk edge - use if clk is set to '0'

Bit 7 of the Byte1 is passed on to TDI/DO before the first clk of TMS and is held static for the duration of TMS clocking. No read operation will take place.

Eighth bit goes to TDI......**always**

Takeaway: *Always* put last bit clocked to TDI in next TMS txfr

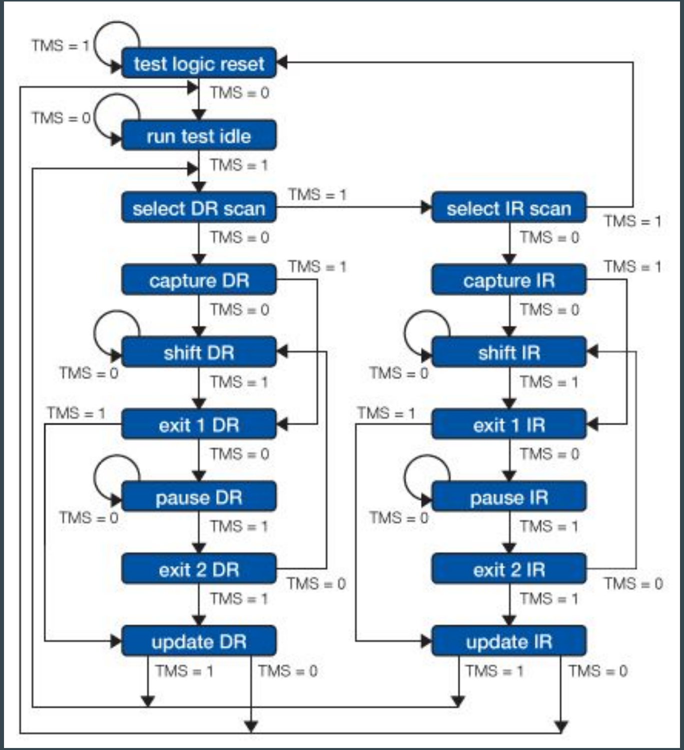
Ex: 8 bits to TDI becomes two cmds:

1. Clock TDI (first seven TDI bits)
2. Clock TMS (last TDI bit)

Screenshot from FTDI MPSSE datasheet

Appendix B: JTAG TAP (Test Access Port)

- State machine for controlling target
- Two registers defined by standard:
  - Instruction Register (IR)
  - Data Register (DR)
- Other registers as defined by implementor
  - E.g. RISC-V debug spec defines DTM Control and Status register

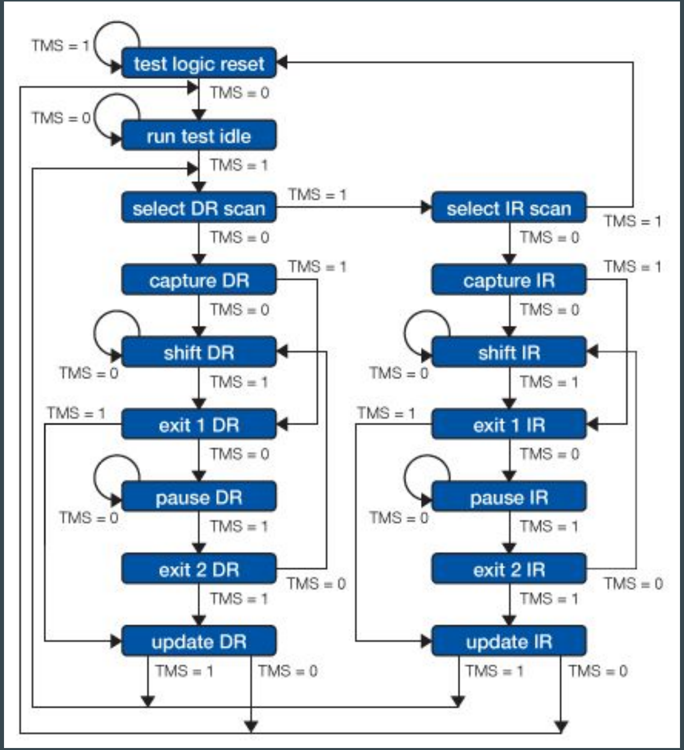


JTAG TAP State Machine

Diagram from https://www.xjtag.com/about-jtag/jtag-a-technical-overview/

Appendix B: JTAG TAP (Test Access Port)

- Four required pins:
  - TCK, TMS, TDI, TDO
- Clocking bits to TMS pin drives state machine
- Good info in RISC-V debug spec Chp 6.



JTAG TAP State Machine

Diagram from https://www.xjtag.com/about-jtag/jtag-a-technical-overview/

JTAG Select Register Example

- **Goal:**
  - Select IDCODE RISC-V DTM (debug transport module) register
- Method:
  - Transfer IDCODE register address to Instruction Register (IR) in DTM TAP

JTAG Select Register Example

- **Assume:**
  - RISC-V target adheres to RISC-V Debug Spec v0.13.2 (See Chp. 6)
  - Has two TAPs
    - DTM & Boundary Scan
  - Each TAP has 5 bit IR length
  - Boundary Scan TAP comes before DTM TAP in the TAP chain
  - Both TAPs start in RUN-TEST-IDLE
  - Both TAP IRs set to BYPASS register (address 0x1F)

JTAG Select Register Example

- Method:
  - Transfer IDCODE register address to the Instruction Register (IR) in DTM TAP
- To transfer to the IR, we will:
  - Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR
  - Transfer IDCODE register address to IR
  - Transition DTM TAP back to RUN-TEST-IDLE
    - Puts TAP in known state for whatever is next

JTAG Select Register Example

- Note: Since there are two TAPs, we must also tell boundary scan TAP to ignore data we send to the DTM TAP DR (data register)
  - Done by selecting the BYPASS register in Boundary Scan IR (instruction reg)
- Note: Probe-rs FTDI impl and this example use MPSSE commands which:
  - Clock data out on neg. clock edge
  - Clock data in on pos. clock edge
  - Clock data in least significant bit (LSB) order

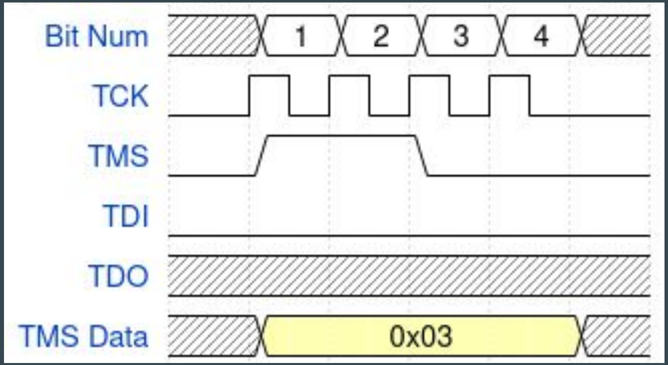
JTAG Select Register Example Steps

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR
2. Transfer IDCODE register address to IR
3. Transition DTM TAP back to RUN-TEST-IDLE

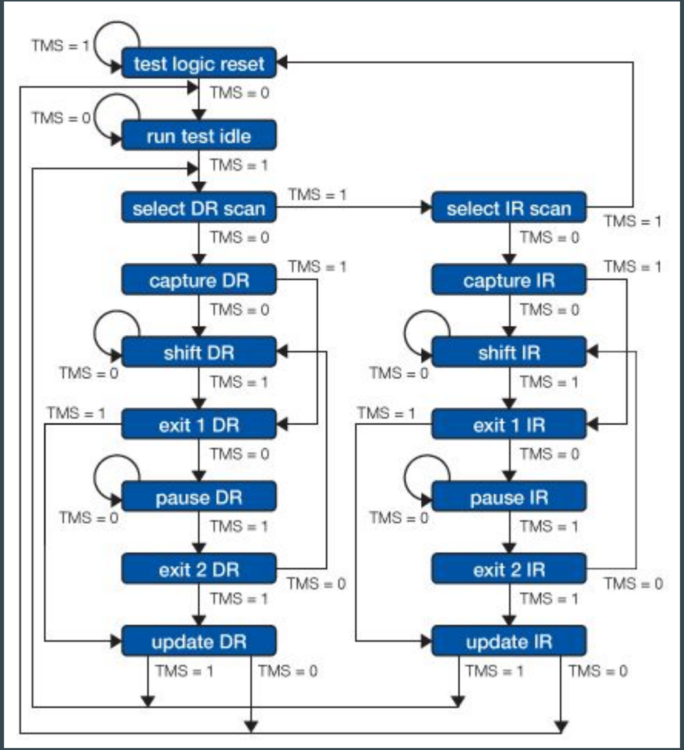
1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

Clock Data to TMS Pin (LSB first, No Read)

4B	04	03
Cmd	Length (bits)	Data (to TMS)



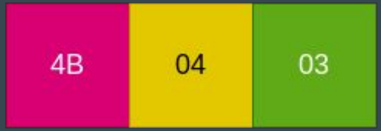
Timing Diagram for Clk Data to TMS Cmd



JTAG TAP State Machine

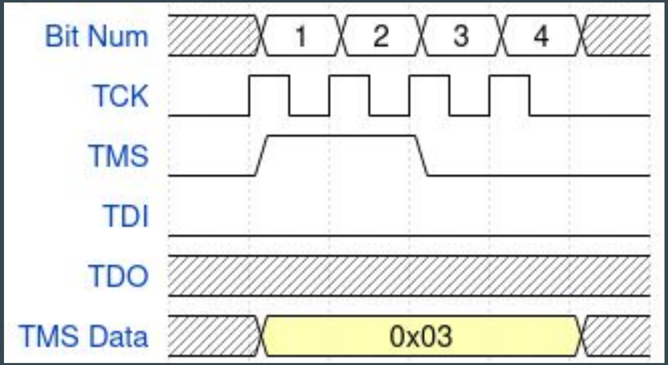
1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

Clock Data to TMS Pin (LSB first, No Read)

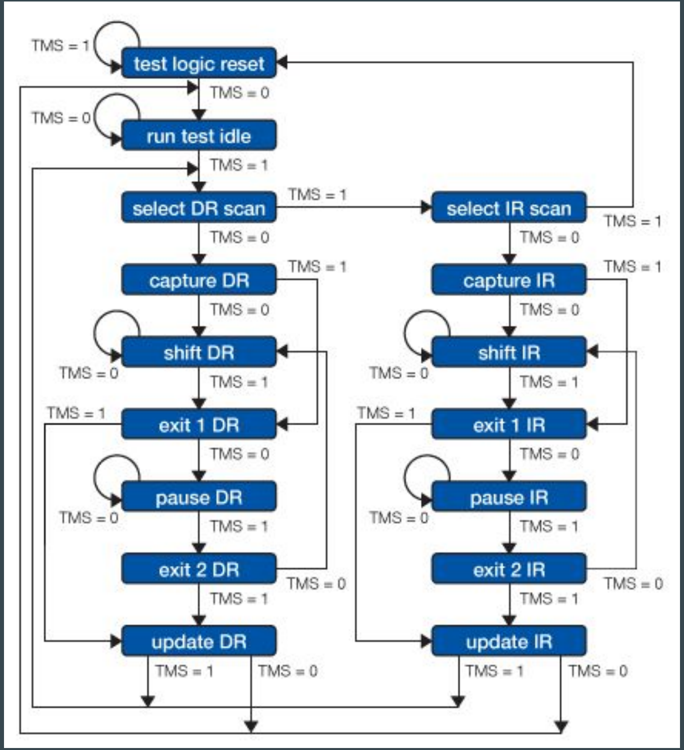


Cmd Length (bits) Data (to TMS)

FTDI MPSSE command to transition from RUN-TEST-IDLE to SHIFT-IR (if start in RUN-TEST-IDLE)



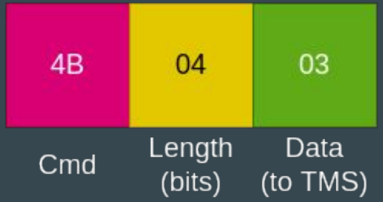
Timing Diagram for Clk Data to TMS Cmd



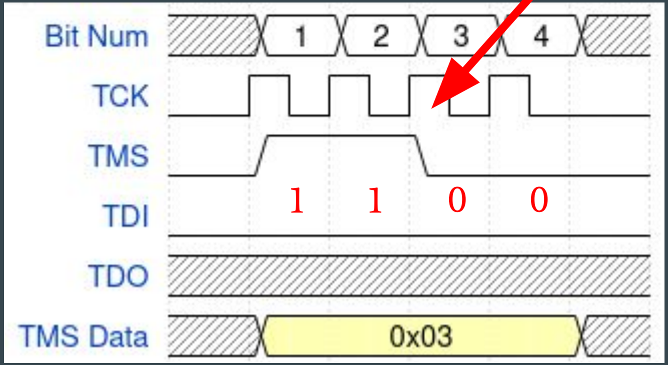
JTAG TAP State Machine

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

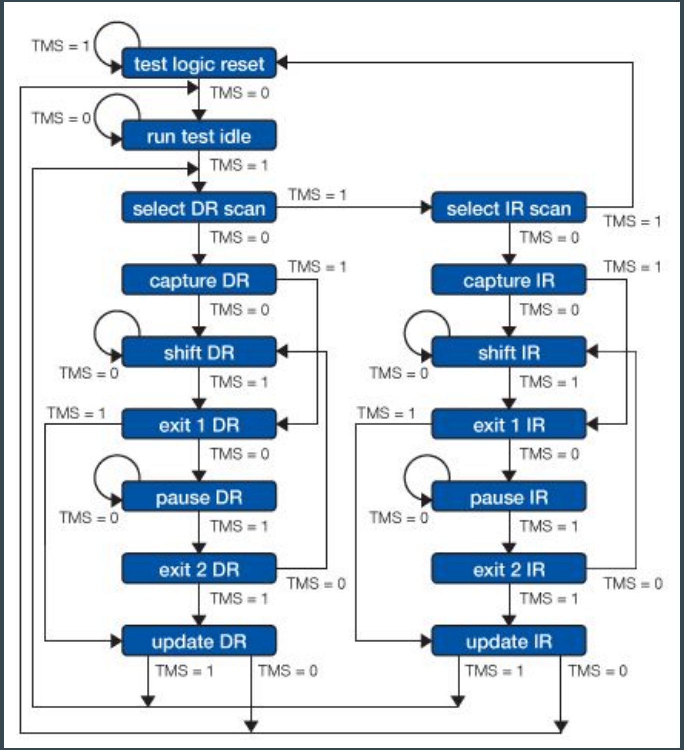
Clock Data to TMS Pin (LSB first, No Read)



Note reversed ordering (i.e. 0b0011 is clocked starting with least signif. bit)



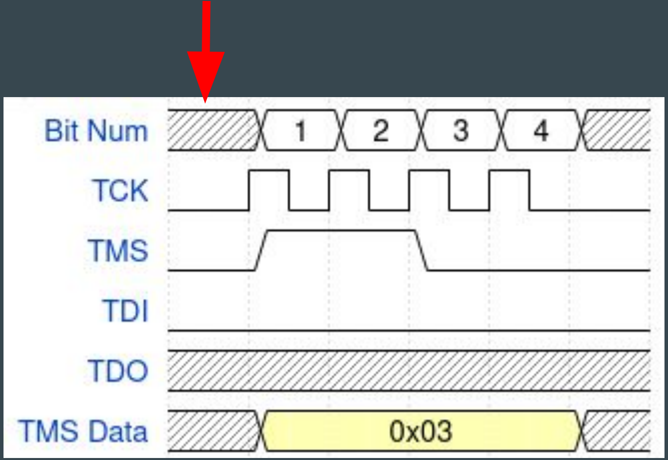
Timing Diagram for Clk Data to TMS Cmd



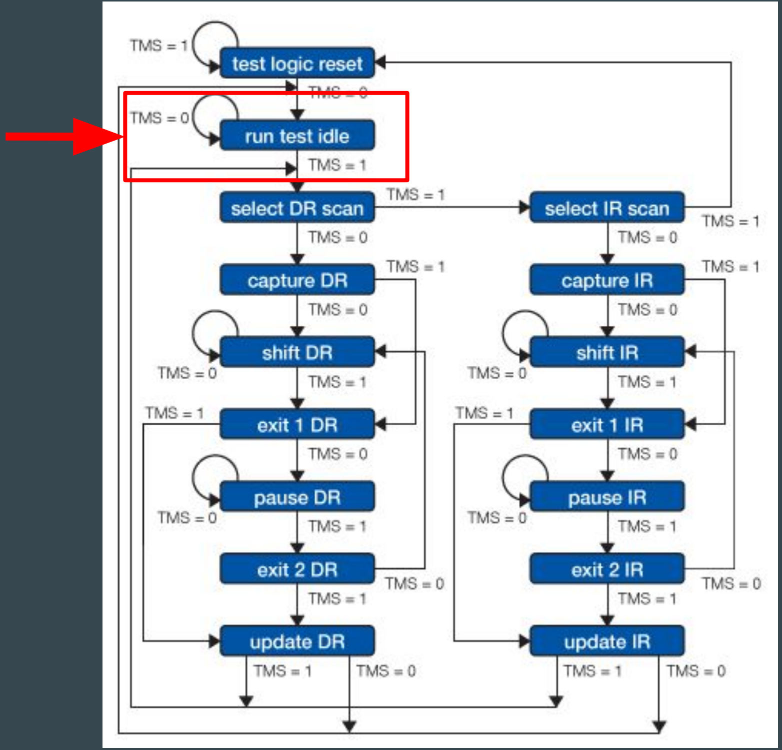
JTAG TAP State Machine

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

Start in RUN-TEST-IDLE



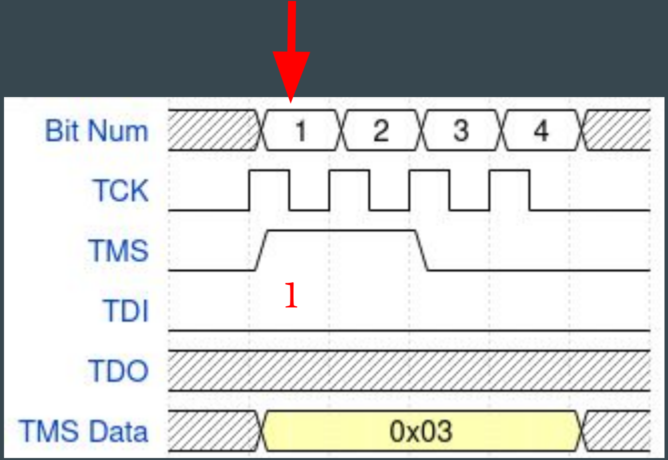
Timing Diagram for Clk Data to TMS Cmd



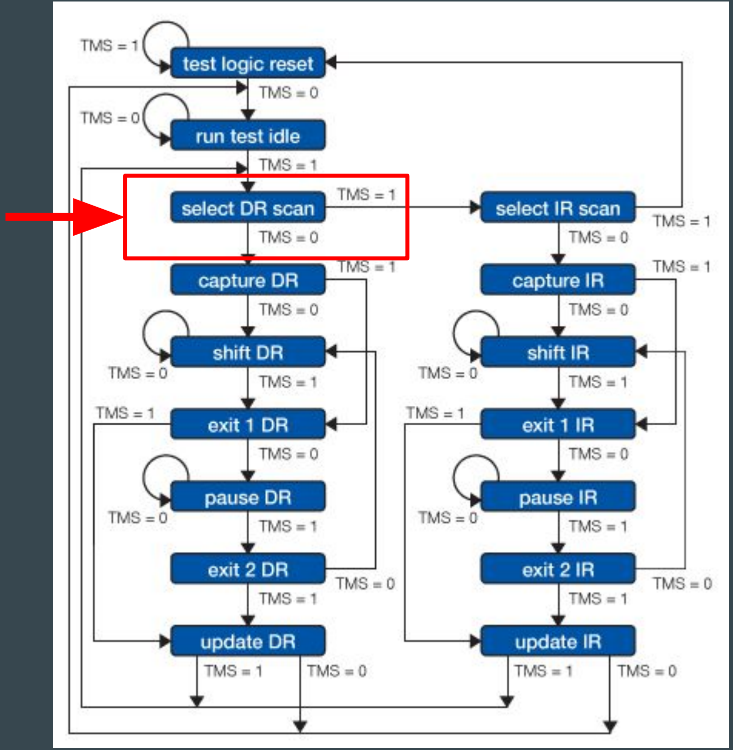
JTAG TAP State Machine

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

TMS bit is 1, go to SELECT-DR-SCAN



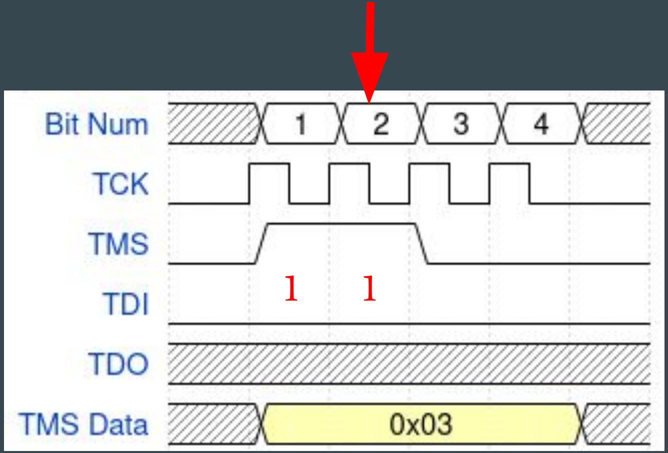
Timing Diagram for Clk Data to TMS Cmd



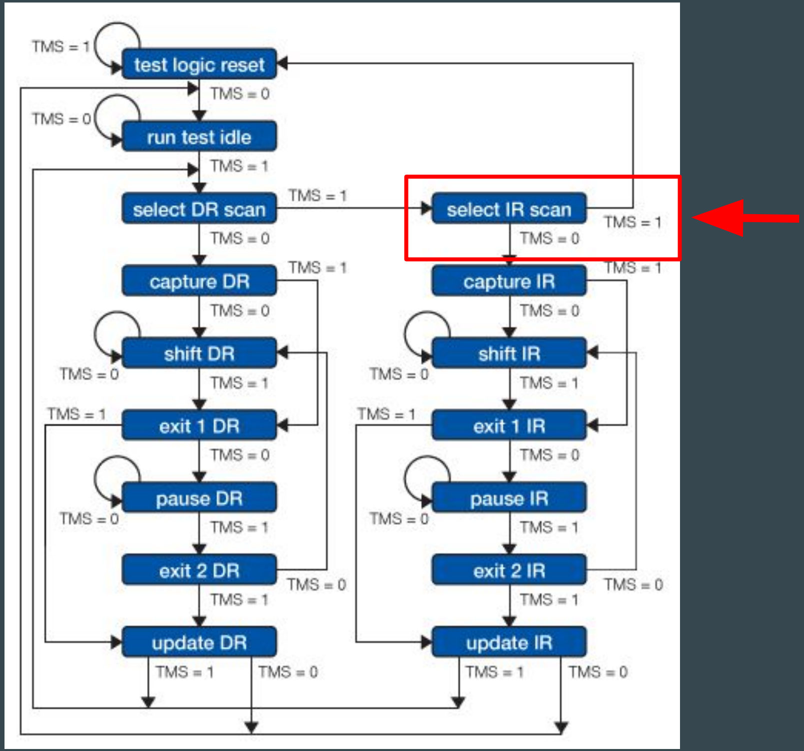
JTAG TAP State Machine

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

TMS bit is 1, go to SELECT-IR-SCAN



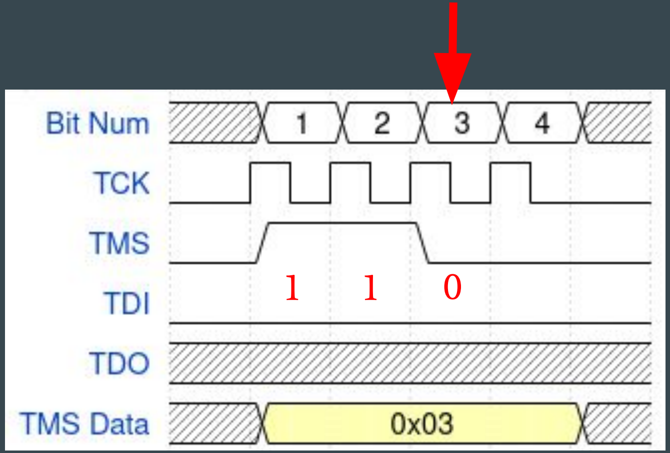
Timing Diagram for Clk Data to TMS Cmd



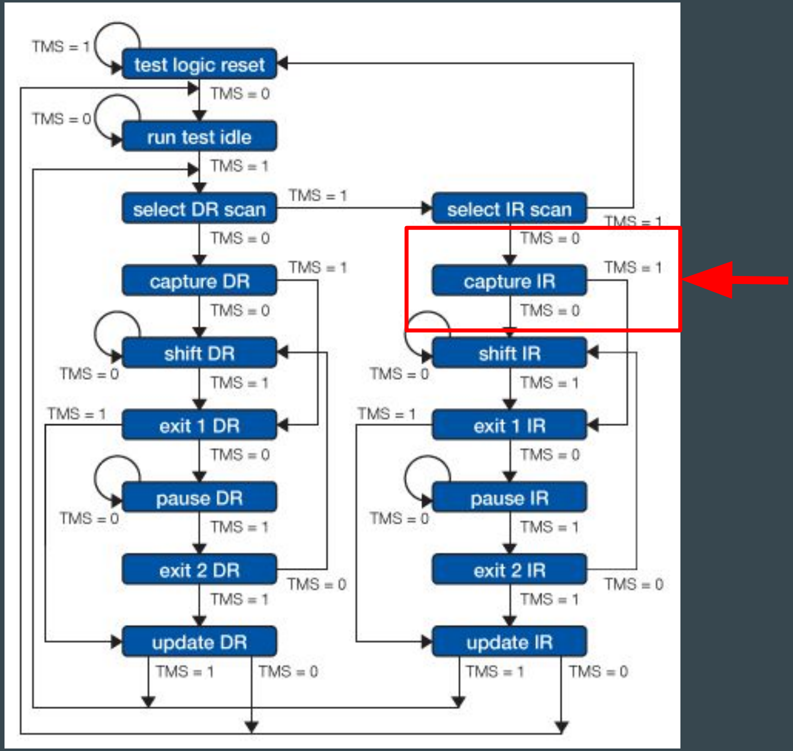
JTAG TAP State Machine

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

TMS bit is 0, go to CAPTURE-IR



Timing Diagram for Clk Data to TMS Cmd

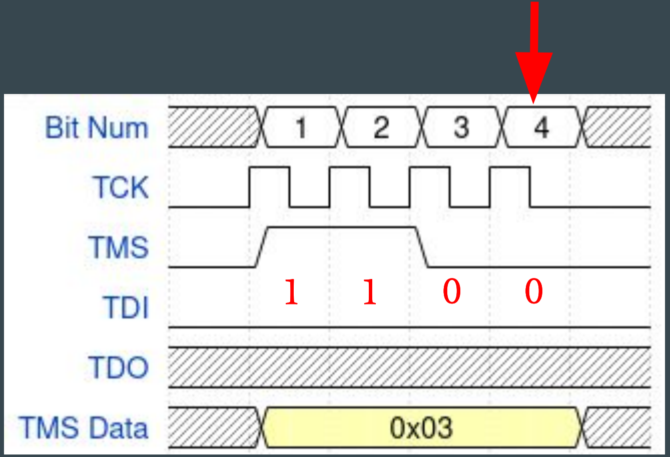


JTAG TAP State Machine

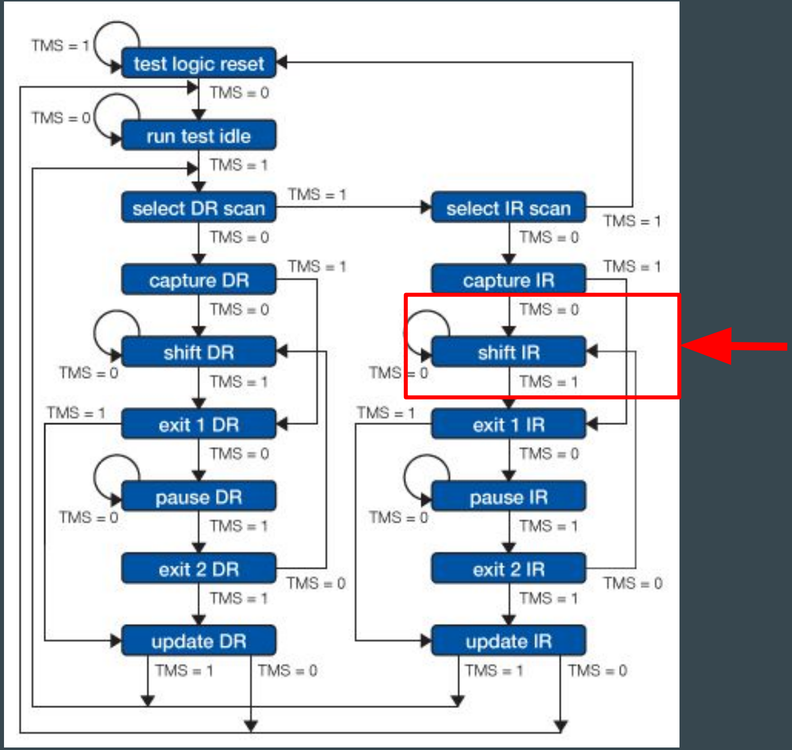
1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR

TMS bit 0, move to SHIFT-IR

(Now can transfer IDCODE instruction using TDI)



Timing Diagram for Clk Data to TMS Cmd



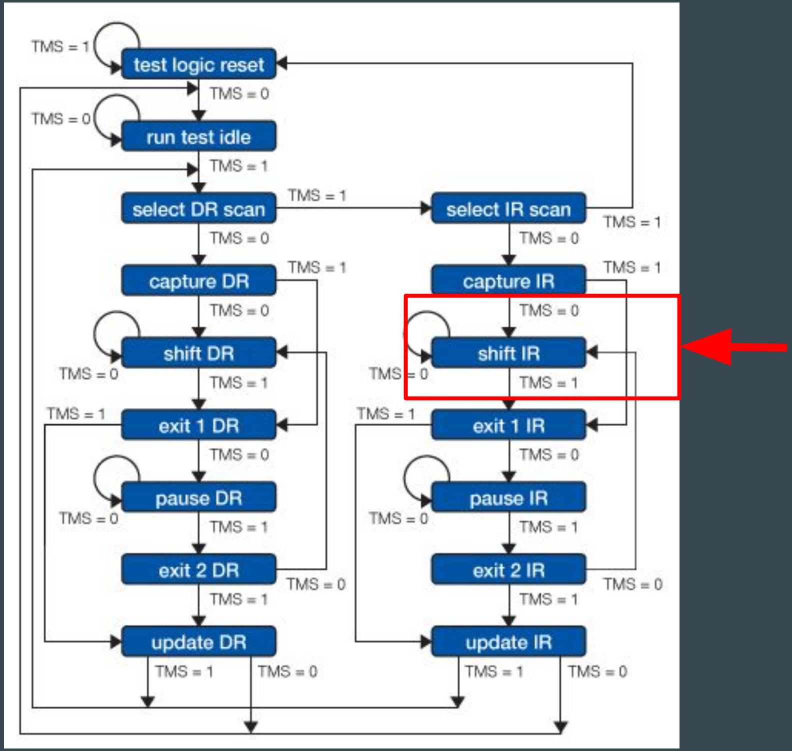
JTAG TAP State Machine

JTAG Select Register Example Steps

1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR
2. Transfer IDCODE register address to IR
3. Transition DTM TAP back to RUN-TEST-IDLE

2. Transfer the IDCODE register address to the IR

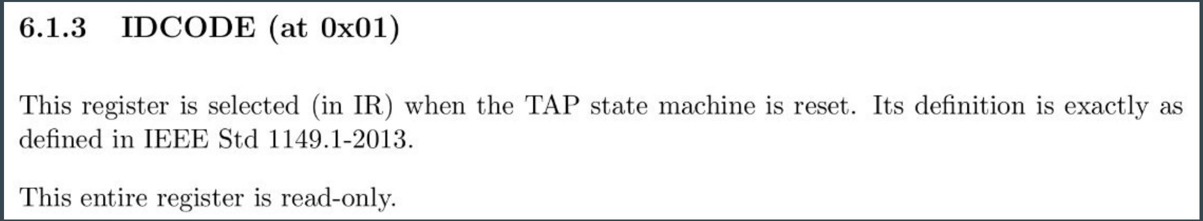
- TAP stays in SHIFT-IR for entire transfer
- When txfring data, must set both TAP IRs
  - Debug Transport Module (DTM) TAP
  - Boundary Scan TAP



JTAG TAP State Machine

2. Transfer the IDCODE register address to the IR

- IDCODE register address: **0x01**

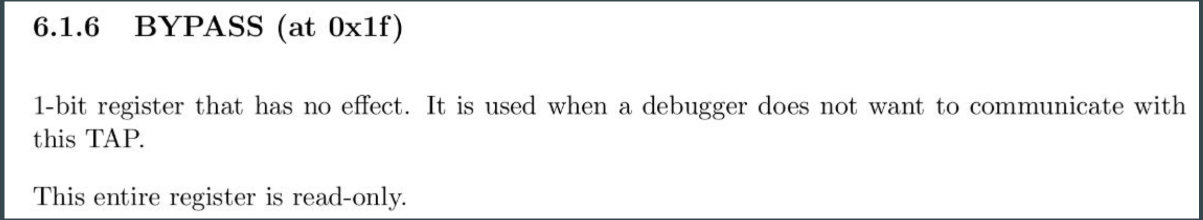


6.1.3 IDCODE (at 0x01)

This register is selected (in IR) when the TAP state machine is reset. Its definition is exactly as defined in IEEE Std 1149.1-2013.

This entire register is read-only.

- BYPASS register address: **0x1F**



6.1.6 BYPASS (at 0x1f)

1-bit register that has no effect. It is used when a debugger does not want to communicate with this TAP.

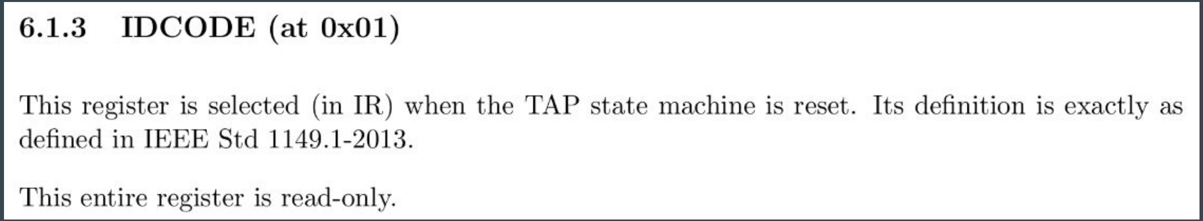
This entire register is read-only.

2. Transfer the IDCODE register address to the IR

- IDCODE register address: **0x01**

↑

Want in DTM IR

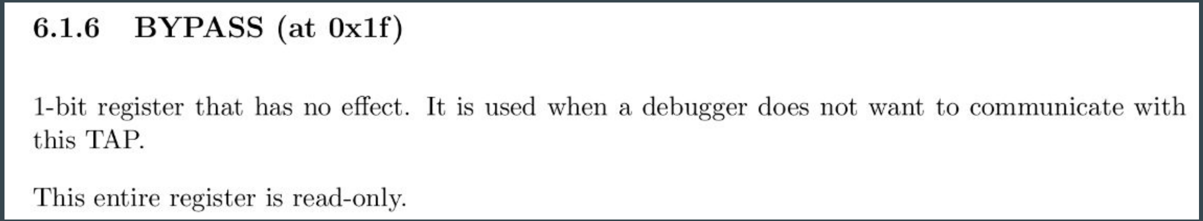


6.1.3 IDCODE (at 0x01)

This register is selected (in IR) when the TAP state machine is reset. Its definition is exactly as defined in IEEE Std 1149.1-2013.

This entire register is read-only.

- BYPASS register address: **0x1F**



6.1.6 BYPASS (at 0x1f)

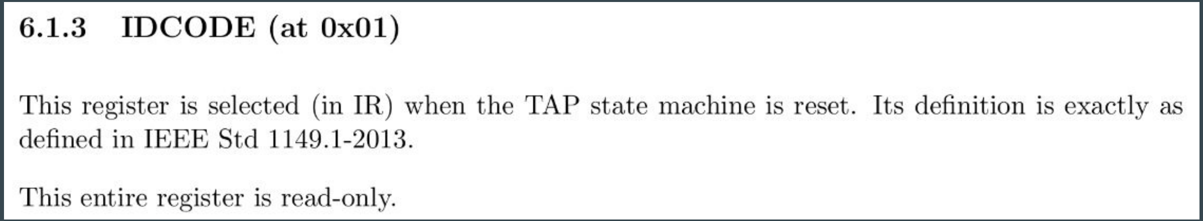
1-bit register that has no effect. It is used when a debugger does not want to communicate with this TAP.

This entire register is read-only.

2. Transfer the IDCODE register address to the IR

- IDCODE register address: **0x01**

Want in DTM IR



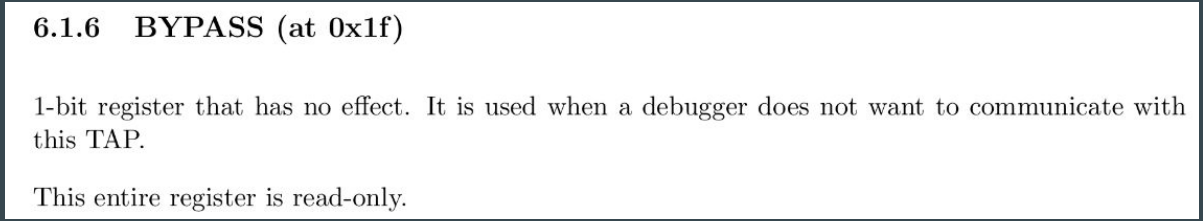
6.1.3 IDCODE (at 0x01)

This register is selected (in IR) when the TAP state machine is reset. Its definition is exactly as defined in IEEE Std 1149.1-2013.

This entire register is read-only.

- BYPASS register address: **0x1F**

Want in Boundary IR (so it ignores data in subseq commands)



6.1.6 BYPASS (at 0x1f)

1-bit register that has no effect. It is used when a debugger does not want to communicate with this TAP.

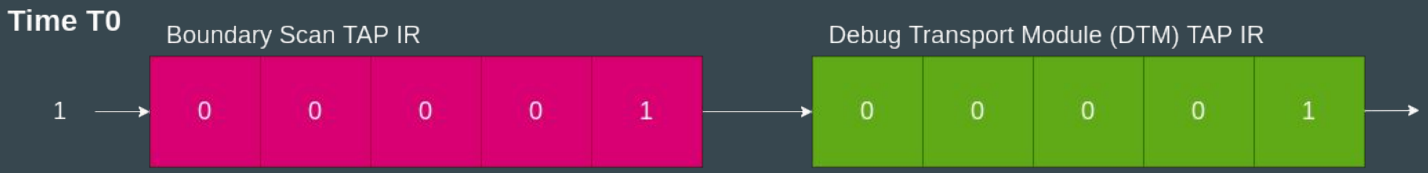
This entire register is read-only.

2. Transfer the IDCODE register address to the IR

- Recall example assumptions:
  - IR length for each TAP is 5 bits
  - Boundary Scan comes before DTM in example TAP chain
- TAP Chains:
  - Bits transferred are shifted into one TAP and out into the following TAP
  - E.g. clock 10 bits in, if Boundary Scan first in chain, Boundary Scan will see DTM's IR bits before they get to DTM IR

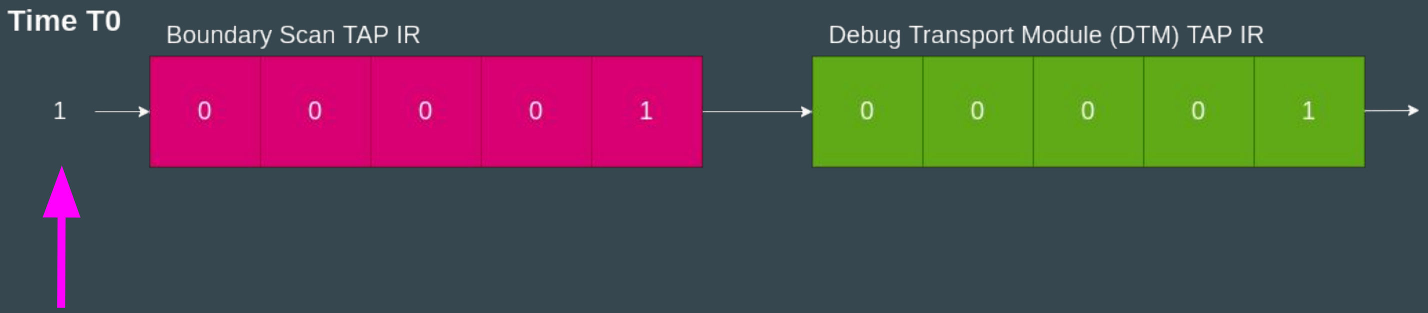
2. Transfer the IDCODE register address to the IR

- TAP Chain Example:



2. Transfer the IDCODE register address to the IR

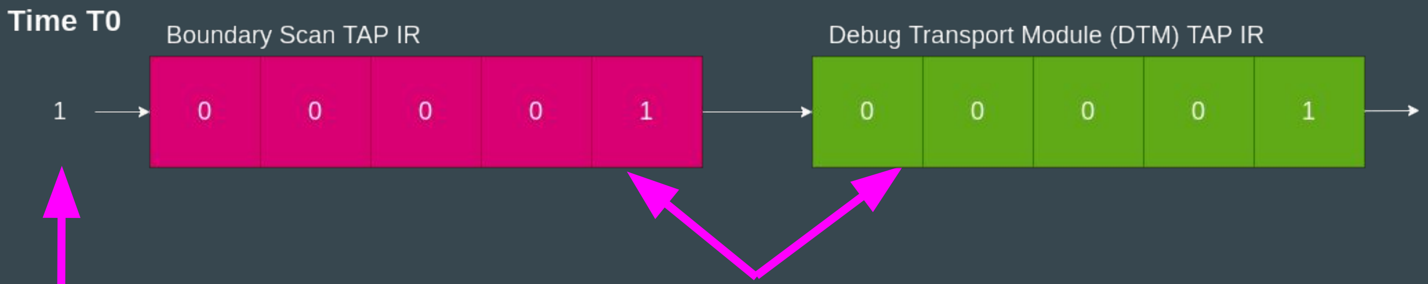
- TAP Chain Example:



About to clock one high bit
into DTM TAP

2. Transfer the IDCODE register address to the IR

- TAP Chain Example:

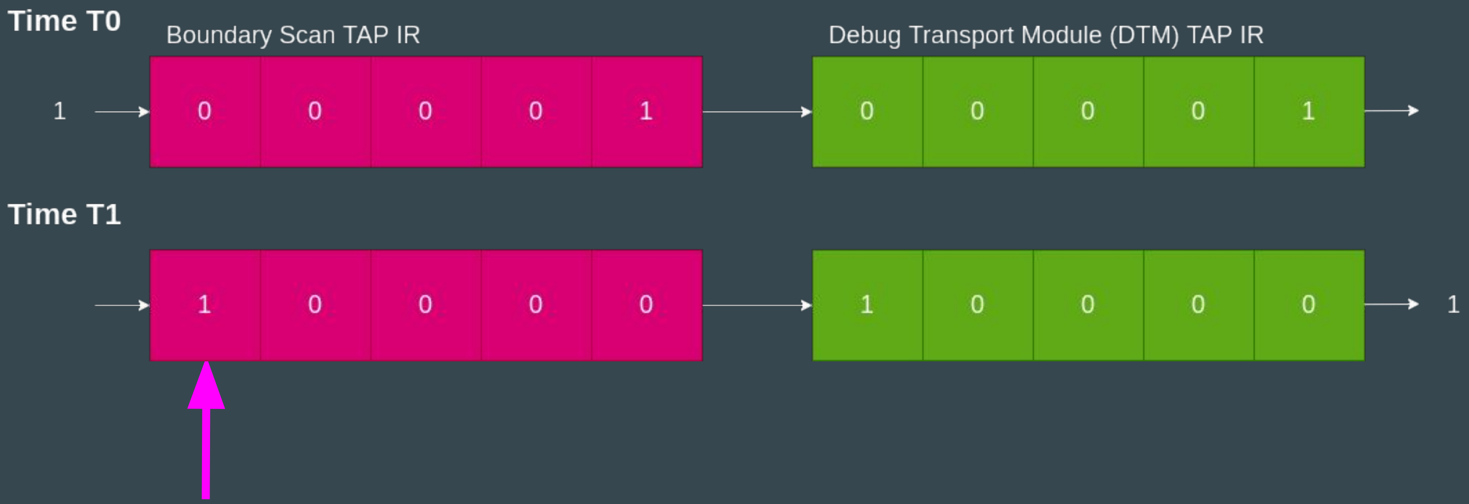


About to clock one high bit into DTM TAP

Note the values currently in these regs

2. Transfer the IDCODE register address to the IR

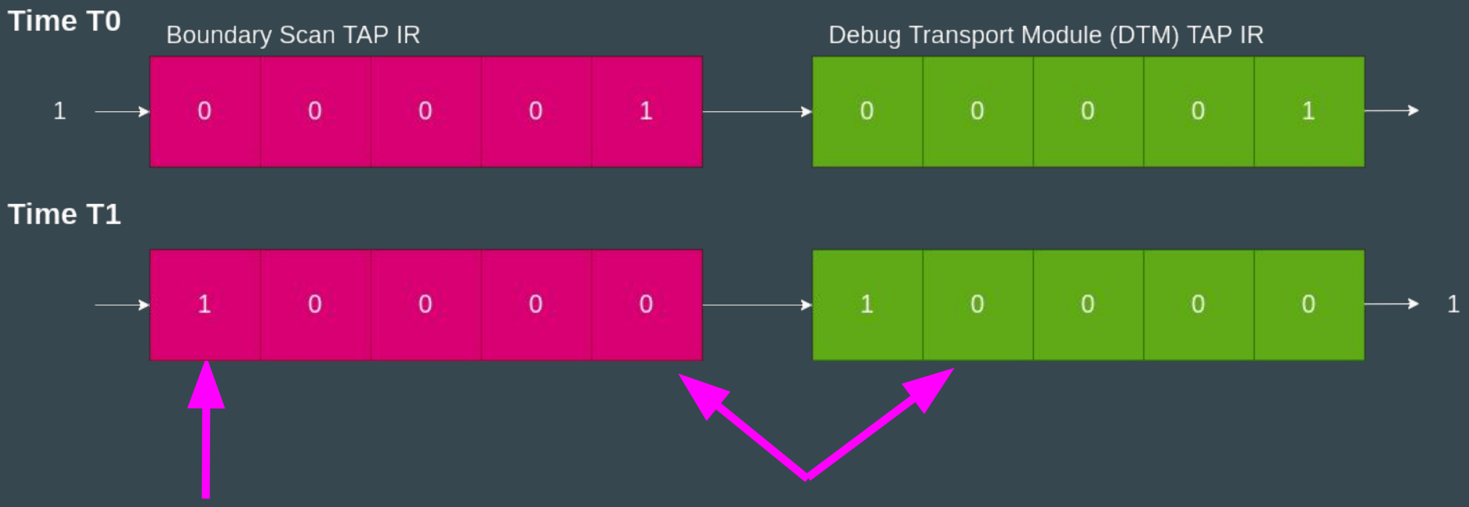
- TAP Chain Example:



Same bit we just clocked in

2. Transfer the IDCODE register address to the IR

- TAP Chain Example:

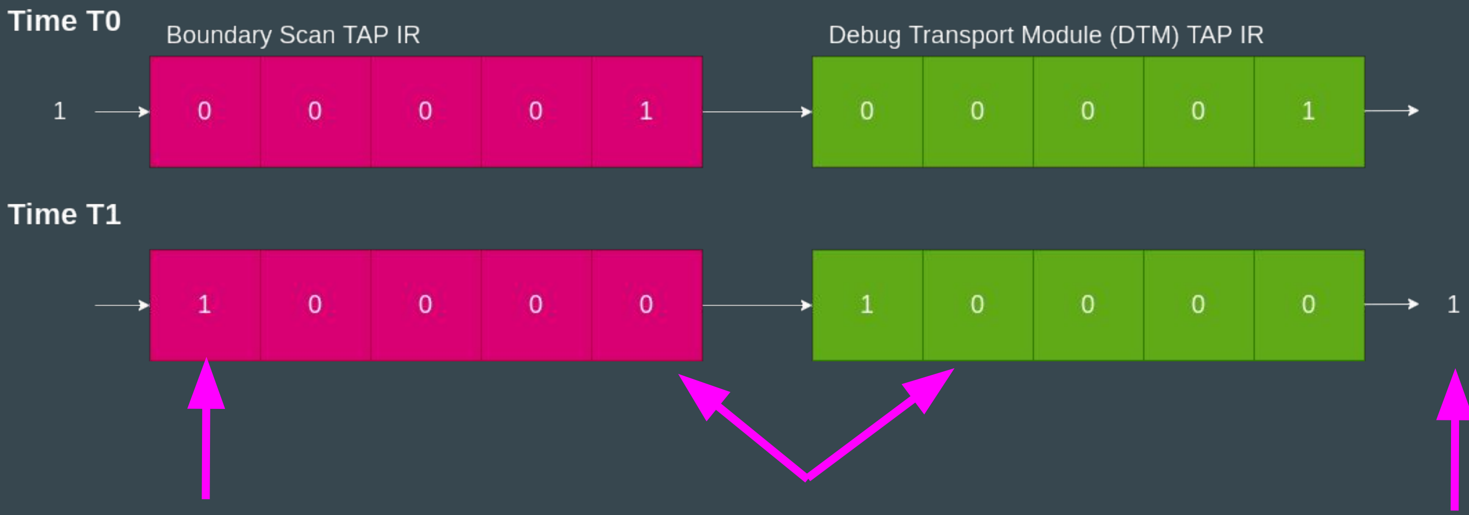


Same bit we just clocked in

Notice that bits in these registers shifted to the right by one to make room

2. Transfer the IDCODE register address to the IR

- TAP Chain Example:



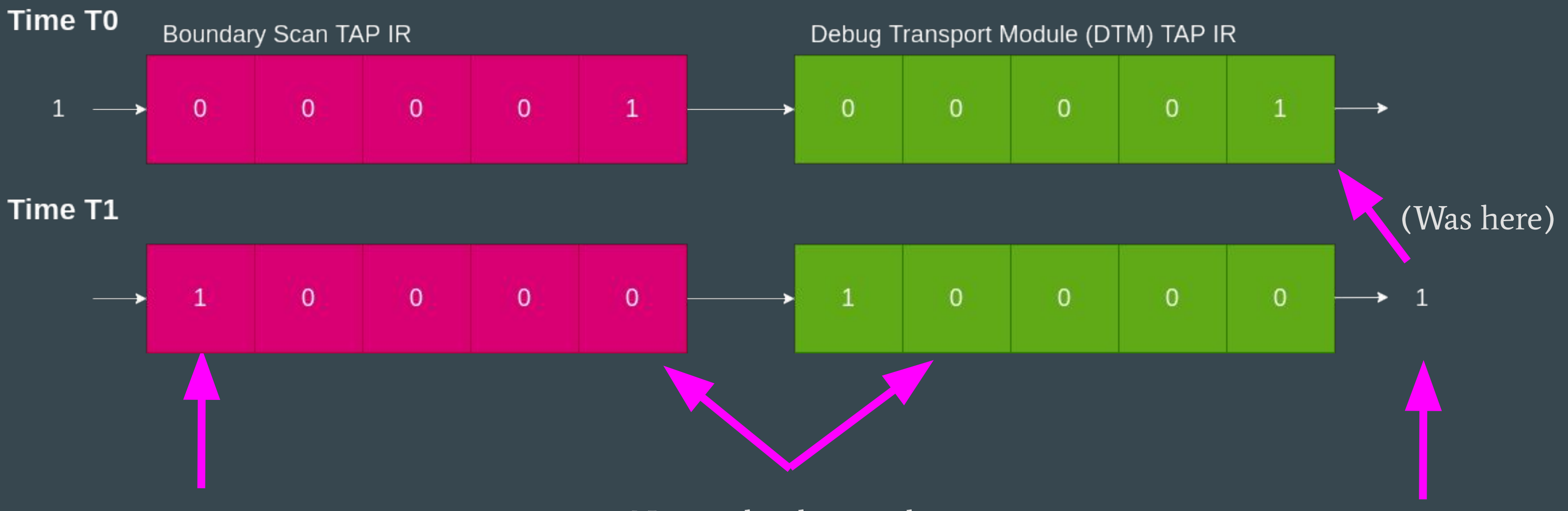
Same bit we just clocked in

Notice that bits in these registers shifted to the right by one to make room

Results in this bit clocked out entirely

2. Transfer the IDCODE register address to the IR

- TAP Chain Example:



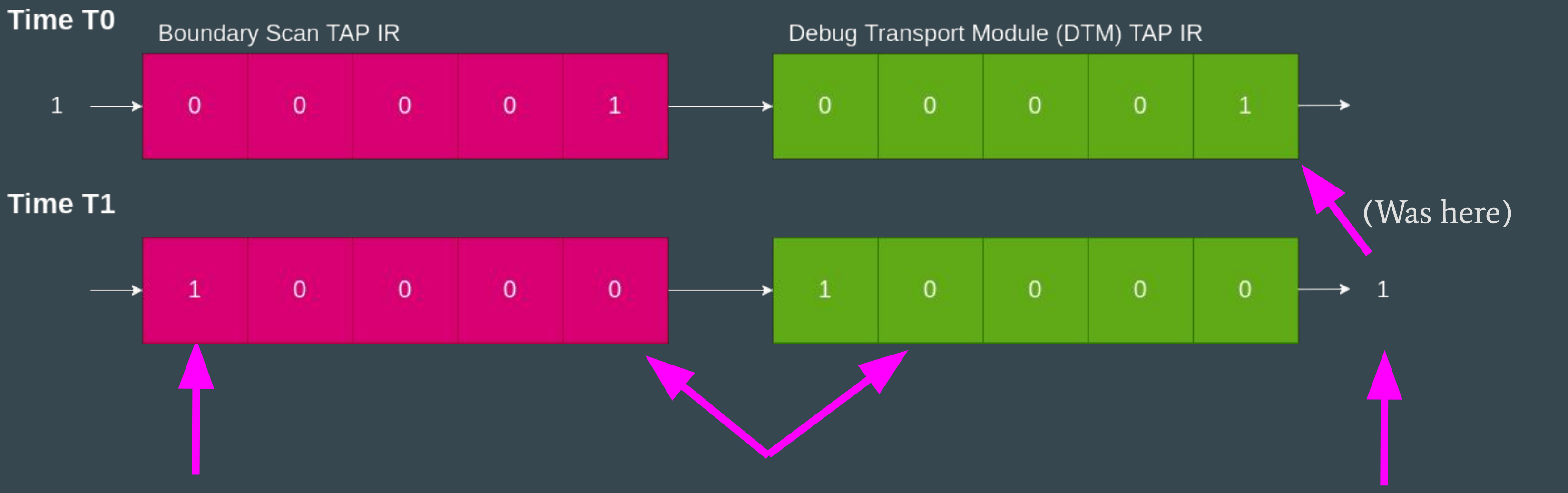
Same bit we just clocked in

Notice that bits in these registers shifted to the right by one to make room

Results in this bit clocked out entirely

2. Transfer the IDCODE register address to the IR

- TAP Chain Example:



Same bit we just clocked in

Notice that bits in these registers shifted to the right by one to make room

Results in this bit clocked out entirely

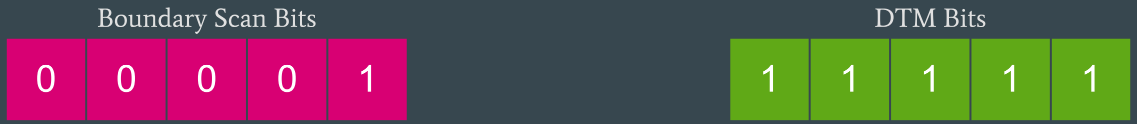
Takeaway: When setting DTM IR, need to *also set Boundary Scan IR*

2. Transfer the IDCODE register address to the IR

- Going back to example, we want:
  - BYPASS (0b1_1111) in the Boundary Scan TAP IR
  - IDCODE (0b0_0001) in the DTM TAP IR

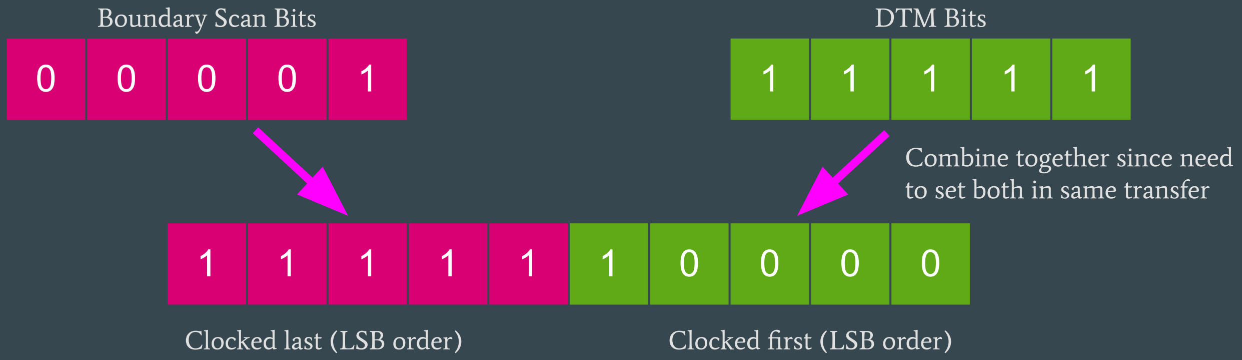
2. Transfer the IDCODE register address to the IR

- Going back to example, we want:
  - BYPASS (0b1_1111) in the Boundary Scan TAP IR
  - IDCODE (0b0_0001) in the DTM TAP IR
- Since Boundary Scan IR comes first in chain, need to clock in DTM IR bits first



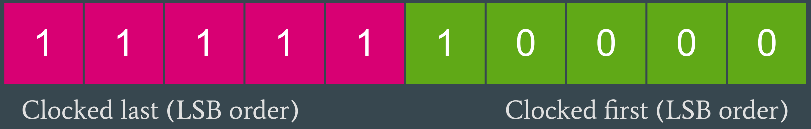
2. Transfer the IDCODE register address to the IR

- Going back to example, we want:
  - BYPASS   (0b1_1111) in the Boundary Scan TAP IR
  - IDCODE (0b0_0001) in the DTM TAP IR
- Since Boundary Scan IR comes first in chain, need to clock in DTM IR bits first



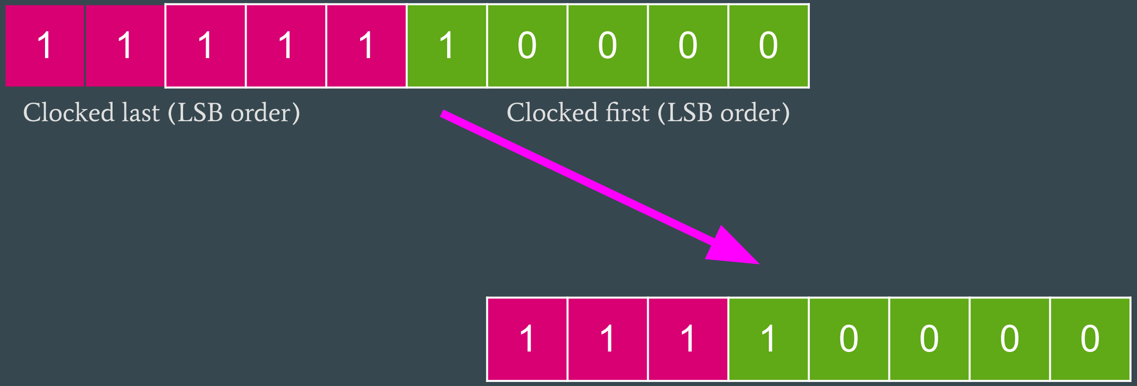
2. Transfer the IDCODE register address to the IR

- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We’re trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands



2. Transfer the IDCODE register address to the IR

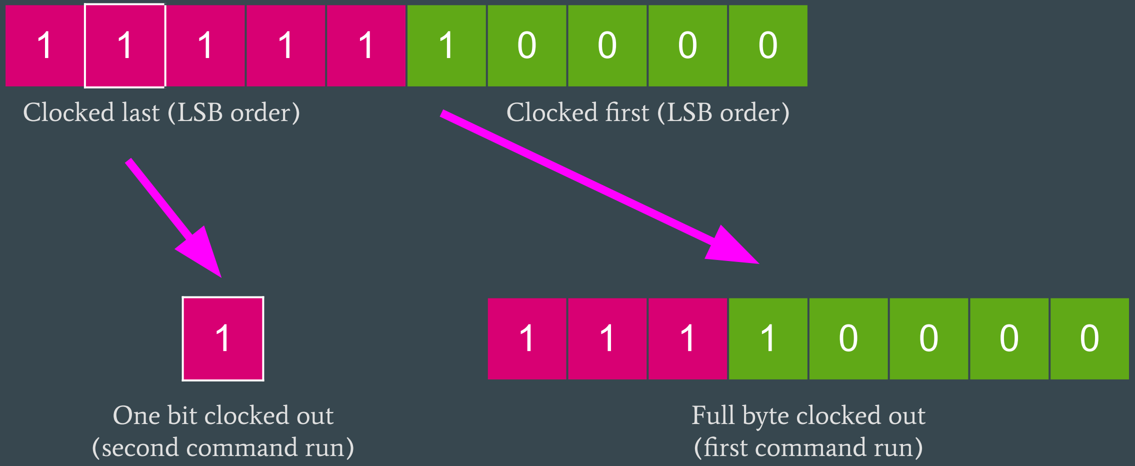
- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We're trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands



Full byte clocked out
(first command run)

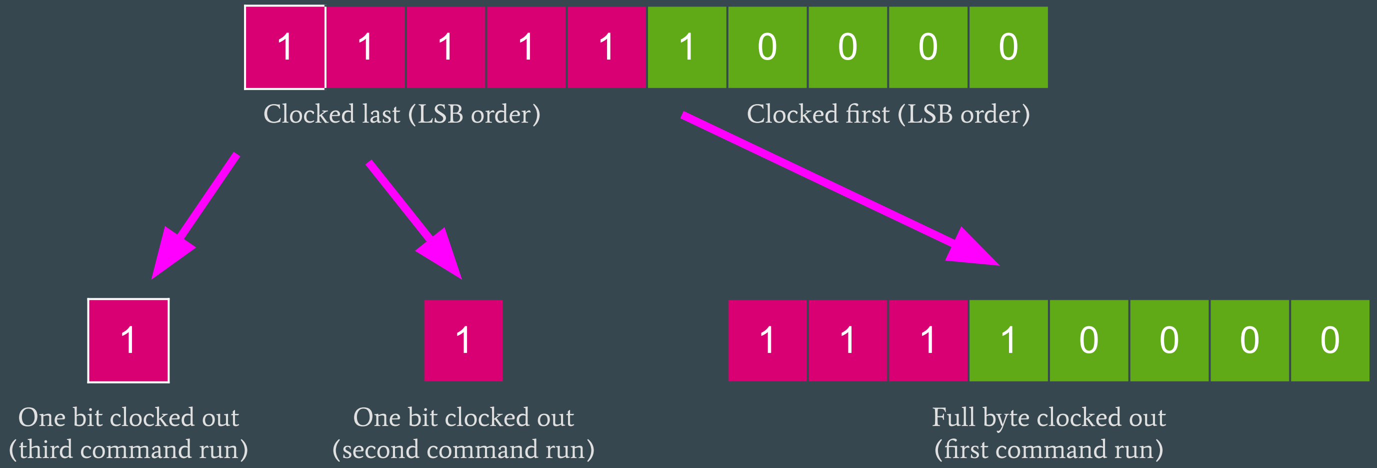
2. Transfer the IDCODE register address to the IR

- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We're trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands



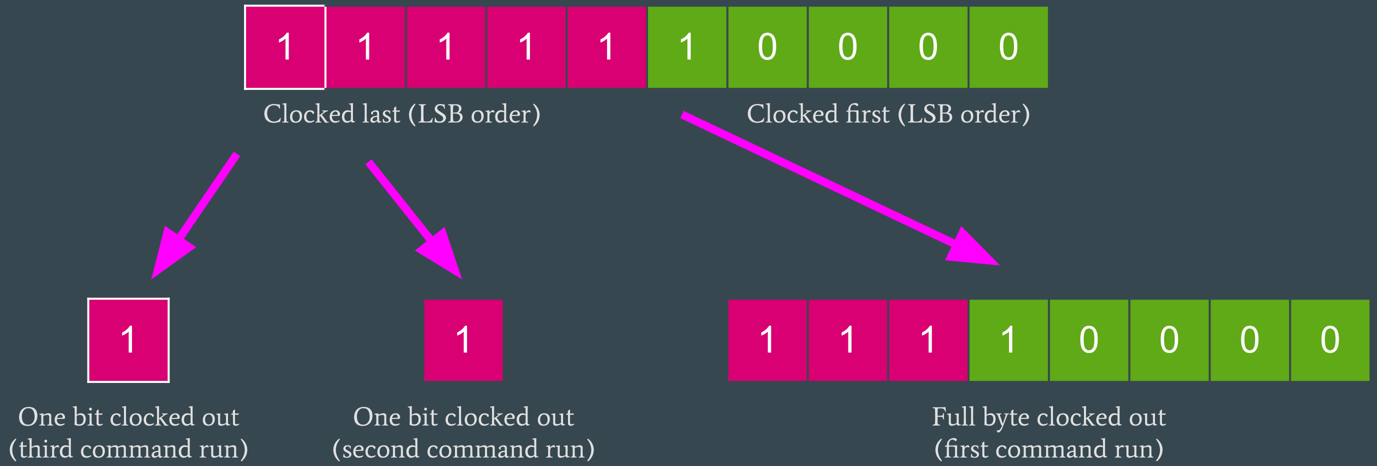
2. Transfer the IDCODE register address to the IR

- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We're trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands


1 1 1 1 1 1 0 0 0 0
Clocked last (LSB order)
Clocked first (LSB order)
1
1
1 1 1 1 0 0 0 0
One bit clocked out (third command run)
One bit clocked out (second command run)
Full byte clocked out (first command run)

2. Transfer the IDCODE register address to the IR

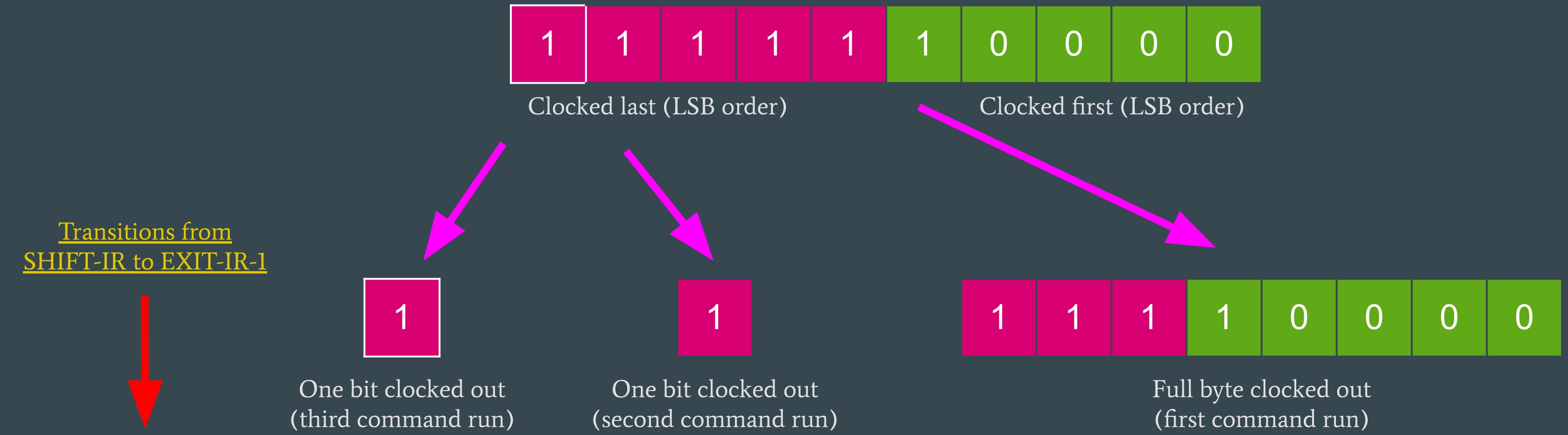
- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We're trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands


1 1 1 1 1 1 0 0 0 0
Clocked last (LSB order)
Clocked first (LSB order)
1
One bit clocked out (third command run)
1
One bit clocked out (second command run)
1 1 1 1 0 0 0 0
Full byte clocked out (first command run)

Note: Last two bits separate cmds because *must always clock last TDI bit in next TMS command*

2. Transfer the IDCODE register address to the IR

- FTDI Limitation:
  - Can only clock bits in form of either full bytes or individual bits (<= 7)
- We're trying to clock out ten bits (each IR is five bits)
- Thus, need to break up bits into separate commands

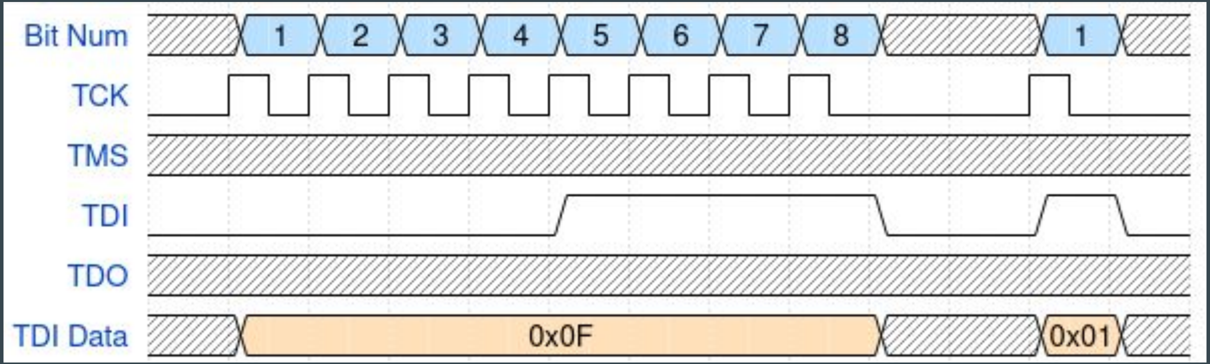


1 1 1 1 1 1 0 0 0 0
Clocked last (LSB order)
Clocked first (LSB order)
Transitions from SHIFT-IR to EXIT-IR-1
1
One bit clocked out (third command run)
1
One bit clocked out (second command run)
1 1 1 1 0 0 0 0
Full byte clocked out (first command run)

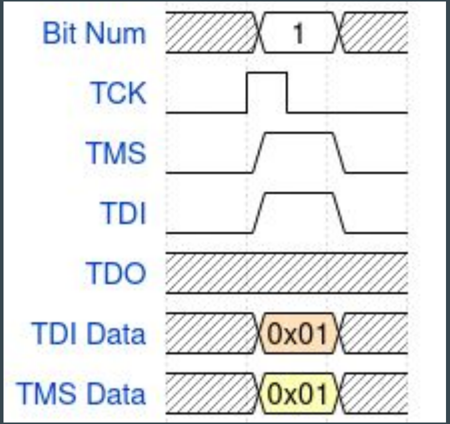
Note: Last two bits separate cmds because *must always clock last TDI bit in next TMS command*

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



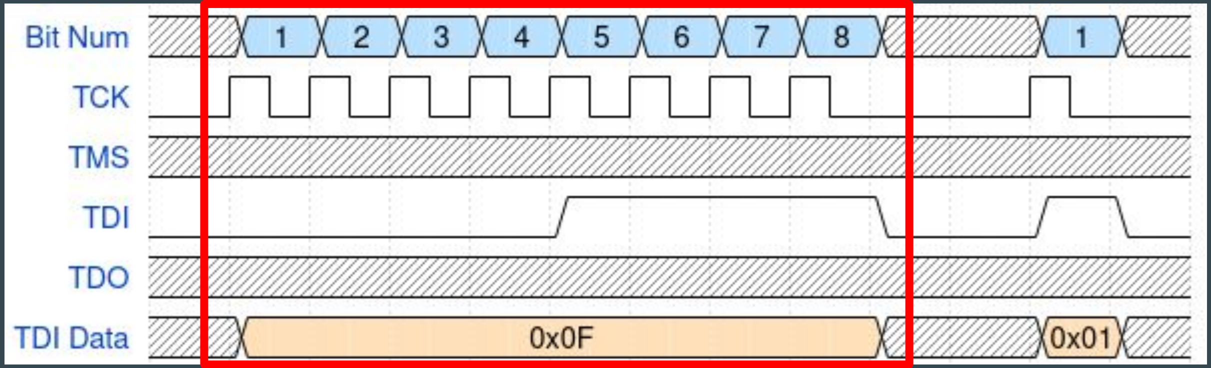
Current IRs

DTM IR				
1	1	1	1	1

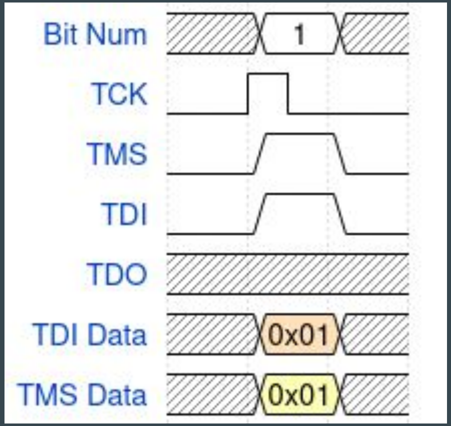
Boundary Scan IR				
1	1	1	1	1

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



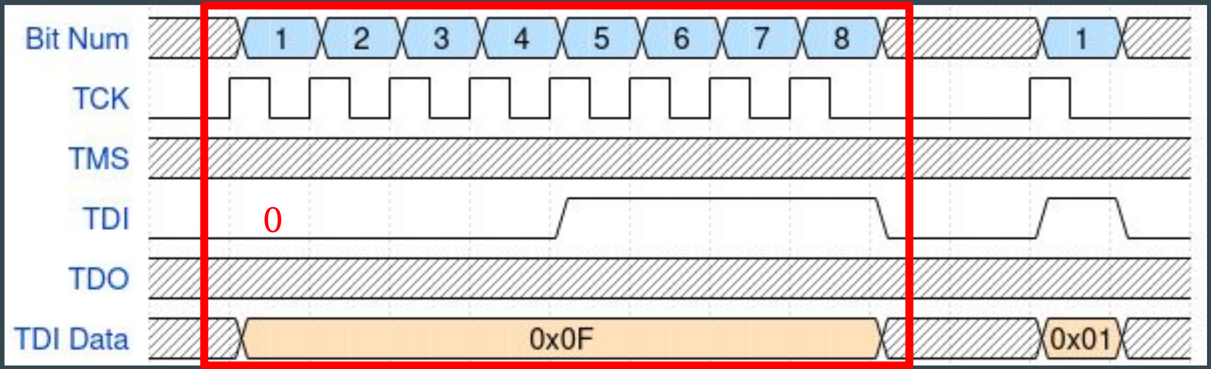
Txfer Byte Cmd

Current IRs

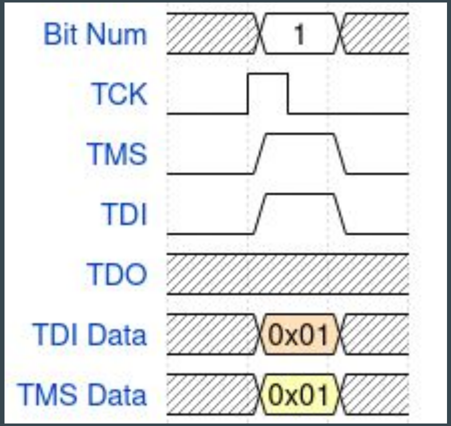
DTM IR					Boundary Scan IR				
1	1	1	1	1	1	1	1	1	1

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds

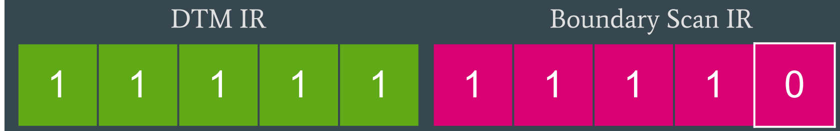


Following Transfer TMS Command



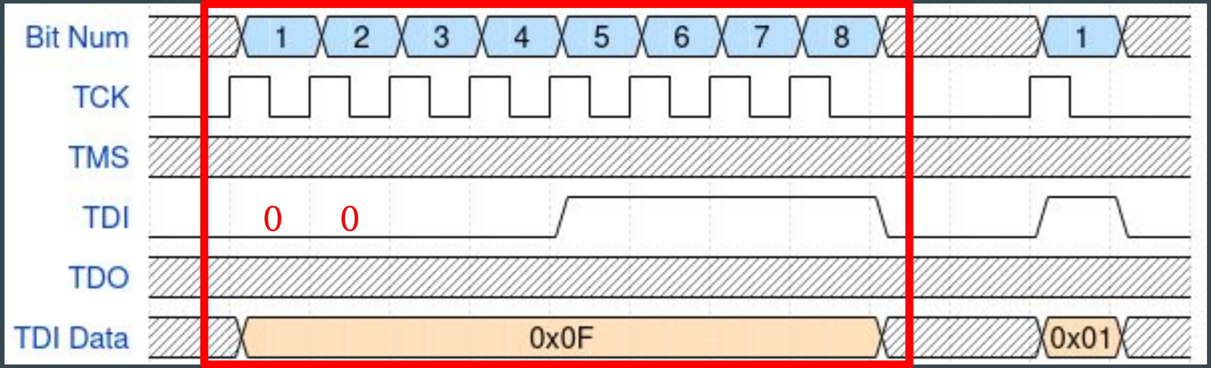
Txfer Byte Cmd

Current IRs



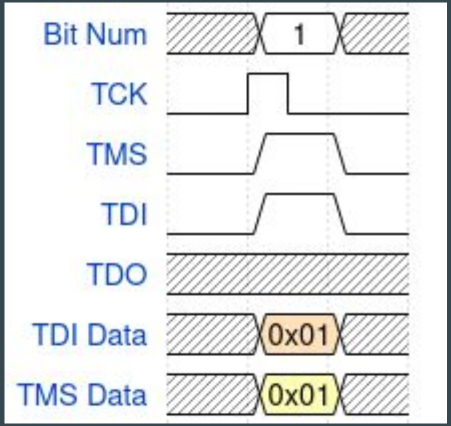
2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds

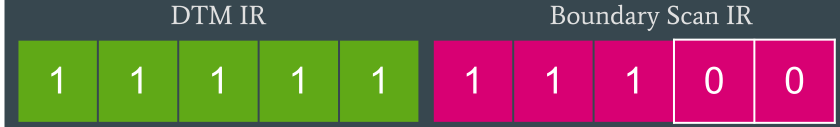


Txfer Byte Cmd

Following Transfer TMS Command

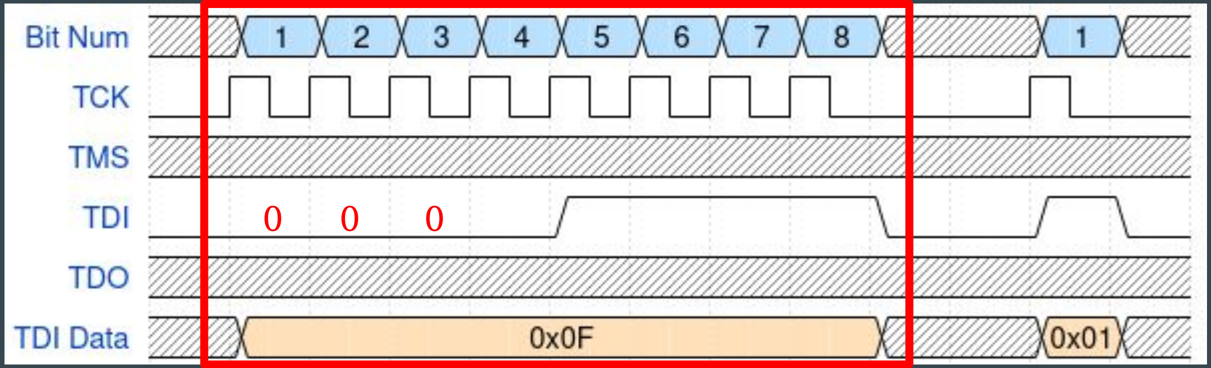


Current IRs

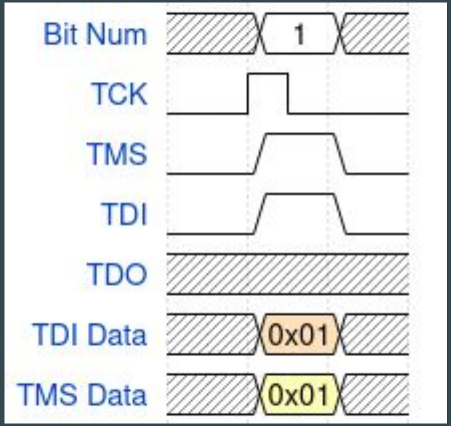


2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



Txfer Byte Cmd

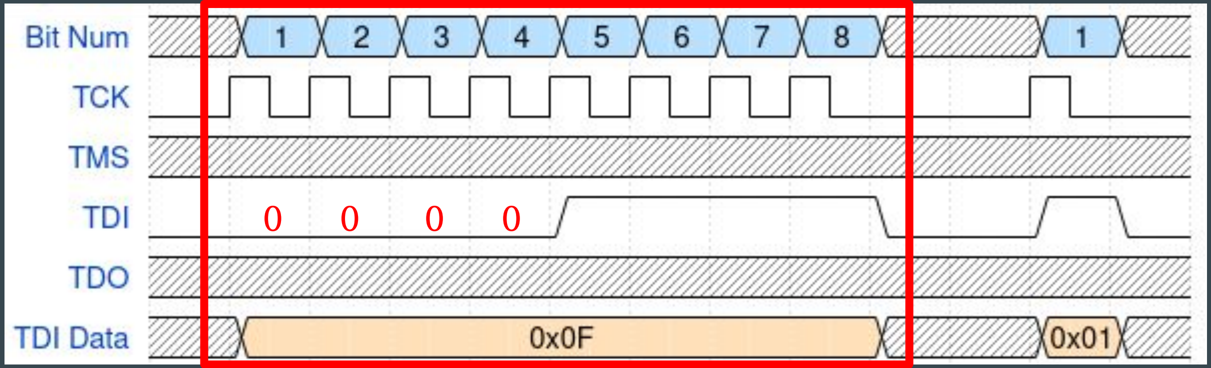
Current IRs

DTM IR				
1	1	1	1	1

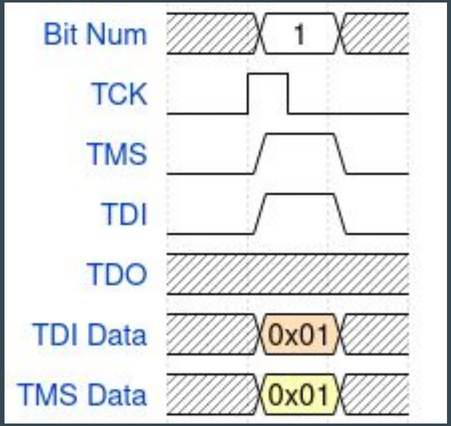
Boundary Scan IR				
1	1	0	0	0

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



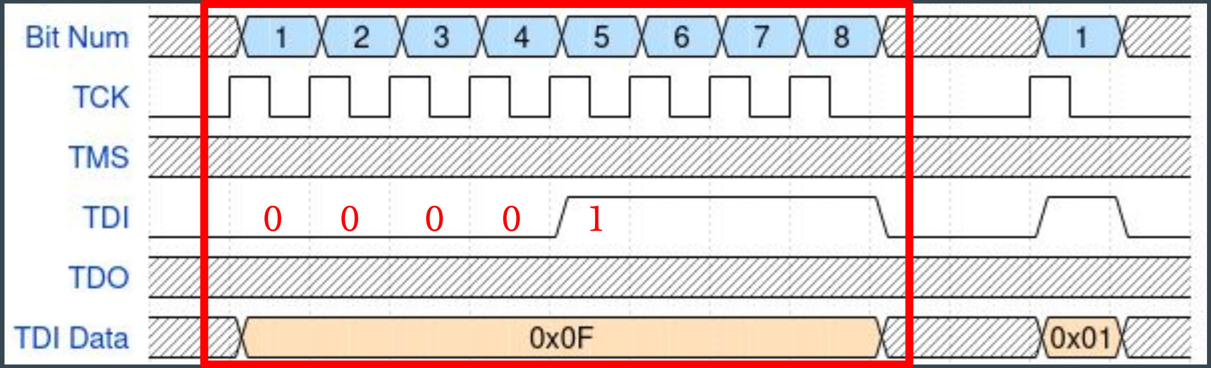
Txfer Byte Cmd

Current IRs

DTM IR					Boundary Scan IR				
1	1	1	1	1	1	0	0	0	0

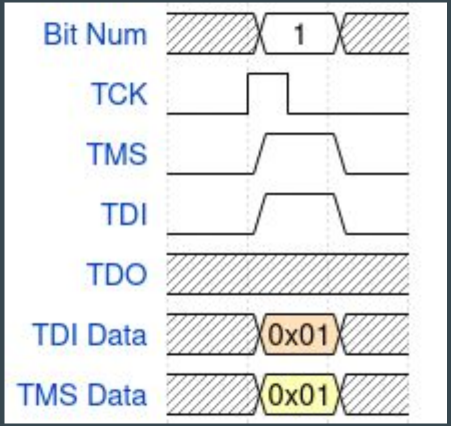
2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Txfer Byte Cmd

Following Transfer TMS Command

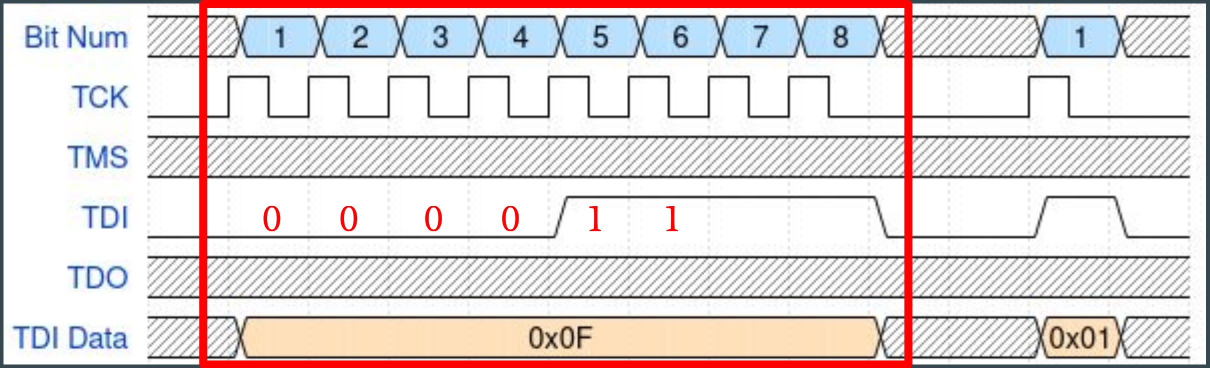


Current IRs

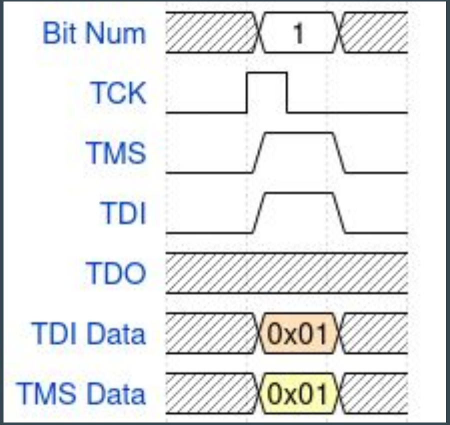
DTM IR					Boundary Scan IR				
1	1	1	1	1	0	0	0	0	1

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



Txfer Byte Cmd

Current IRs

DTM IR

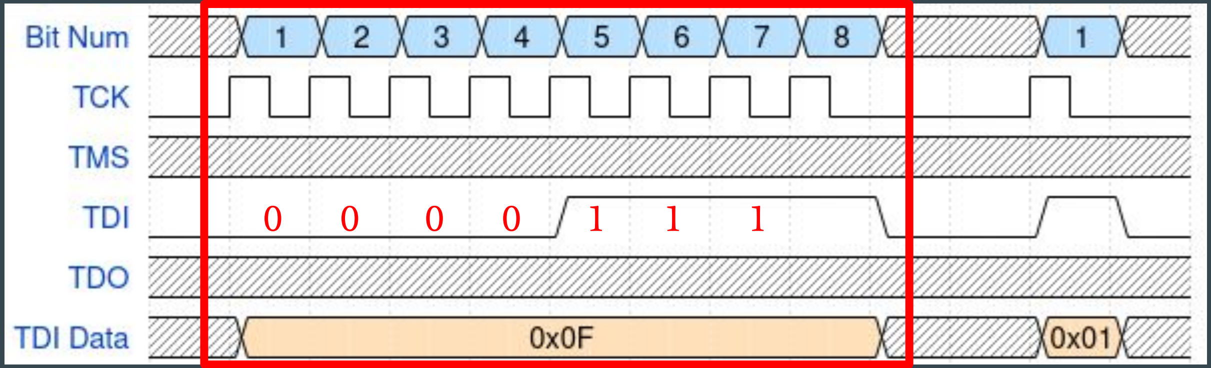
1	1	1	1	0

Boundary Scan IR

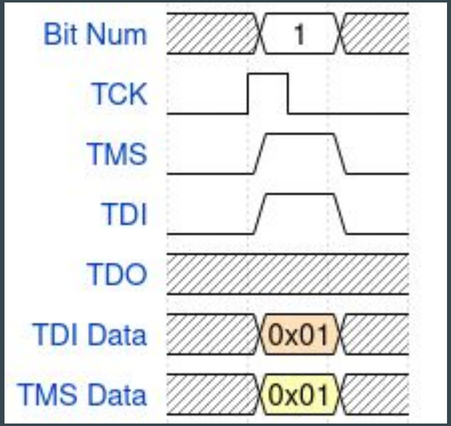
0	0	0	1	1

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



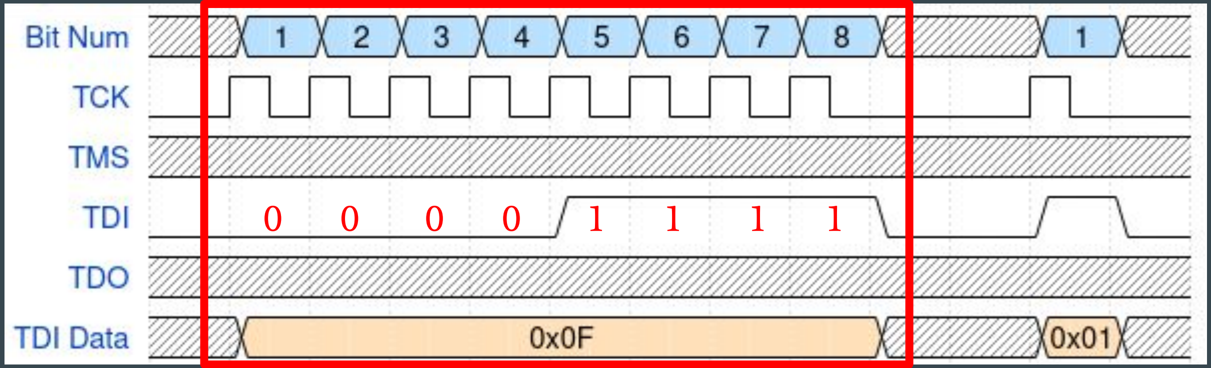
Txfer Byte Cmd

Current IRs

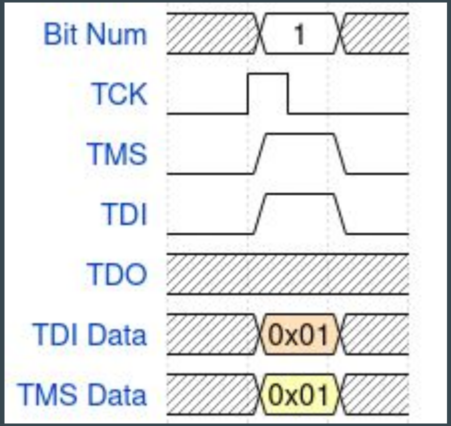
DTM IR					Boundary Scan IR				
1	1	1	0	0	0	0	1	1	1

2. Transfer the IDCODE register address to the IR

First Two Transfer TDI Cmds



Following Transfer TMS Command



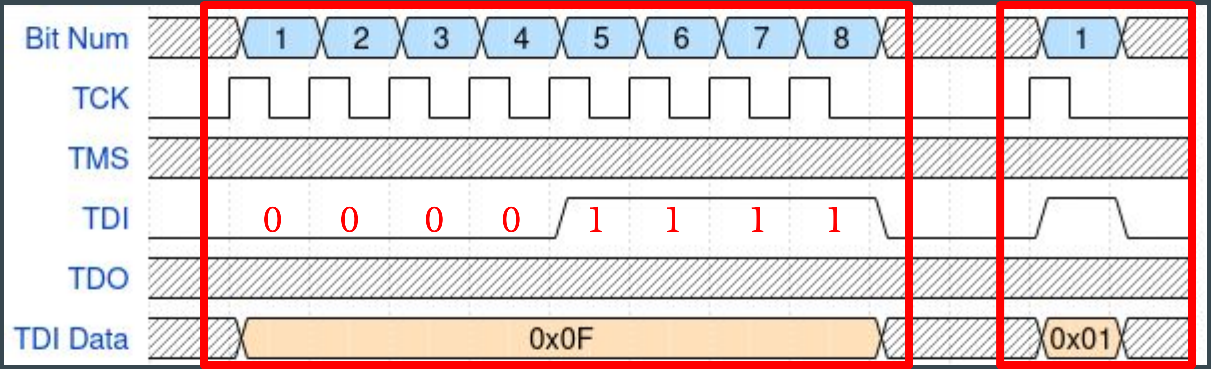
Txfer Byte Cmd

Current IRs

DTM IR					Boundary Scan IR				
1	1	0	0	0	0	1	1	1	1

2. Transfer the IDCODE register address to the IR

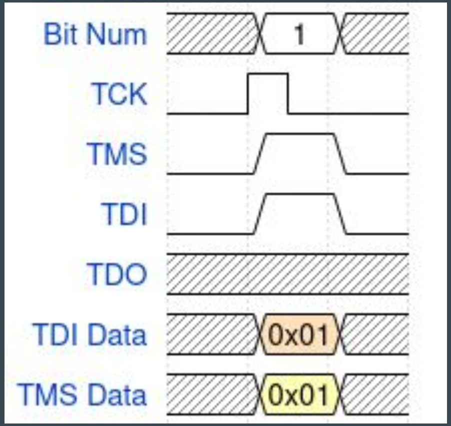
First Two Transfer TDI Cmds



Txfer Byte Cmd

Txfer Bit Cmd

Following Transfer TMS Command

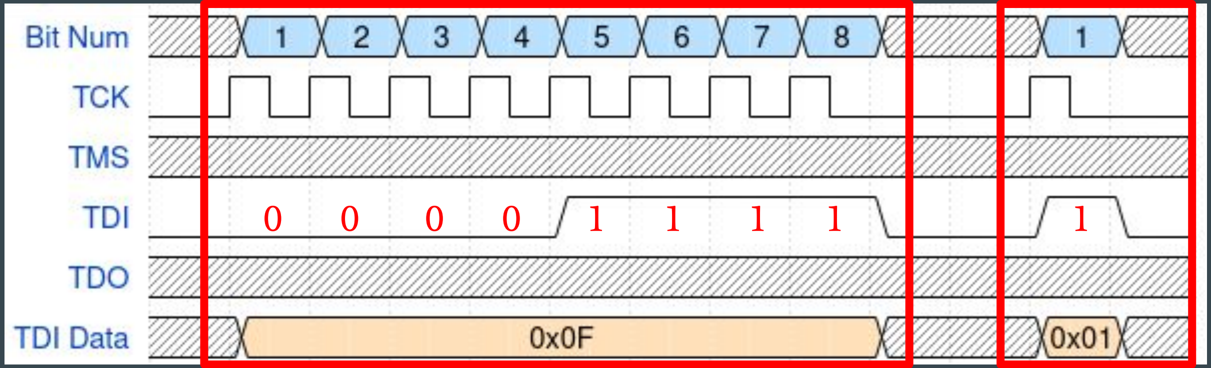


Current IRs

DTM IR					Boundary Scan IR				
1	1	0	0	0	0	1	1	1	1

2. Transfer the IDCODE register address to the IR

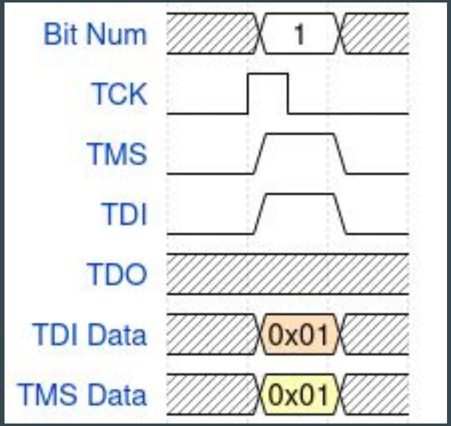
First Two Transfer TDI Cmds



Txfer Byte Cmd

Txfer Bit Cmd

Following Transfer TMS Command



Current IRs

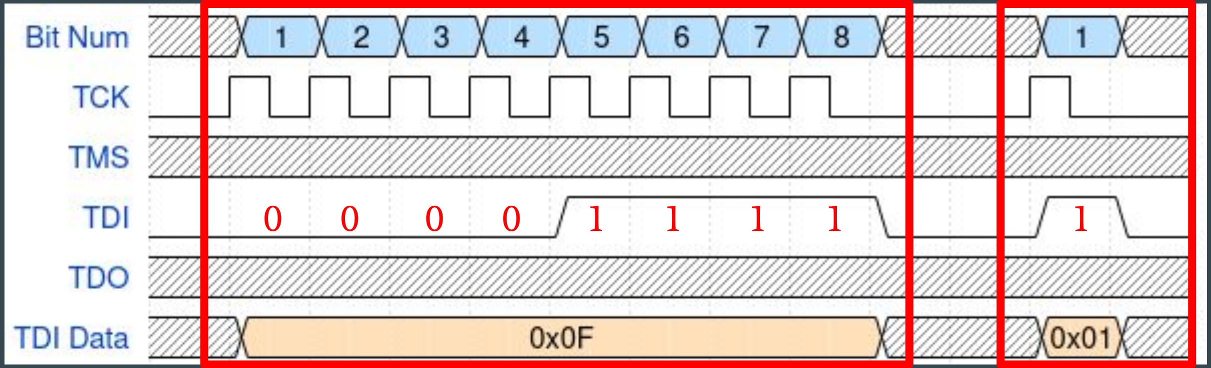
DTM IR

Boundary Scan IR

1	0	0	0	0	1	1	1	1	1

2. Transfer the IDCODE register address to the IR

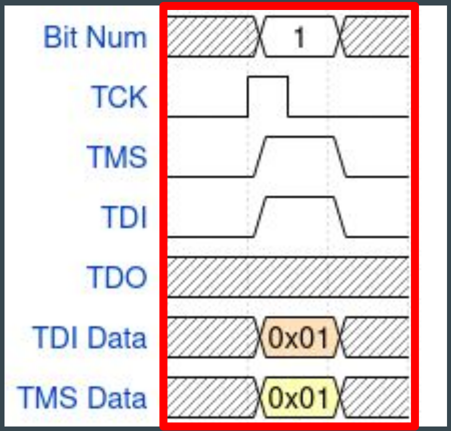
First Two Transfer TDI Cmds



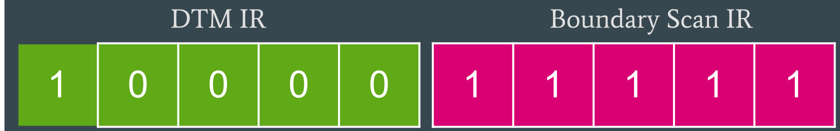
Txfer Byte Cmd

Txfer Bit Cmd

Following Transfer TMS Command

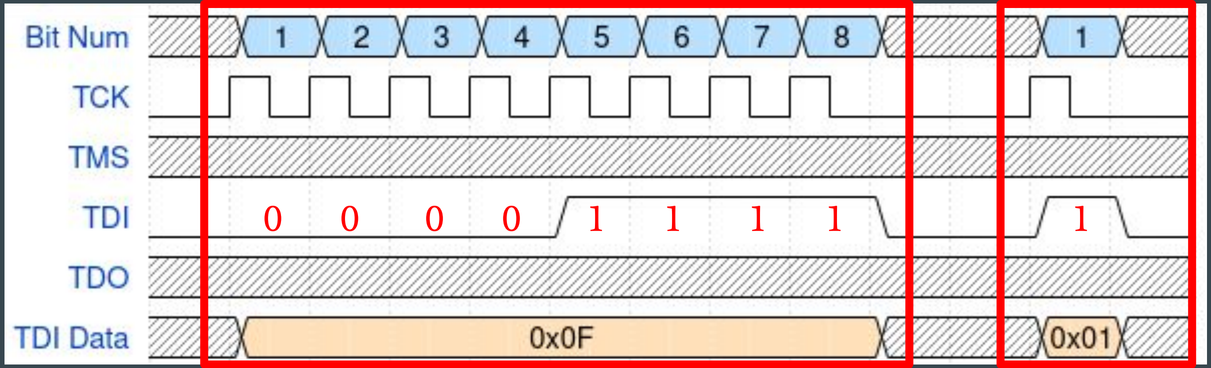


Current IRs



2. Transfer the IDCODE register address to the IR

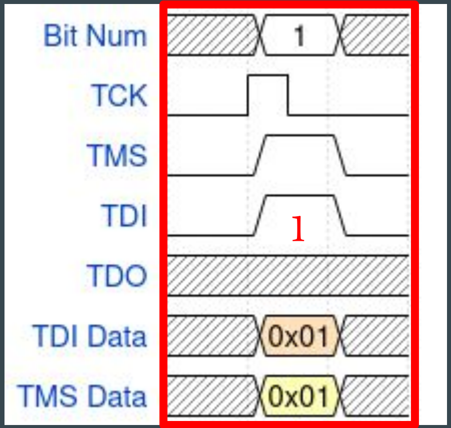
First Two Transfer TDI Cmds



Txfer Byte Cmd

Txfer Bit Cmd

Following Transfer TMS Command



Current IRs

DTM IR

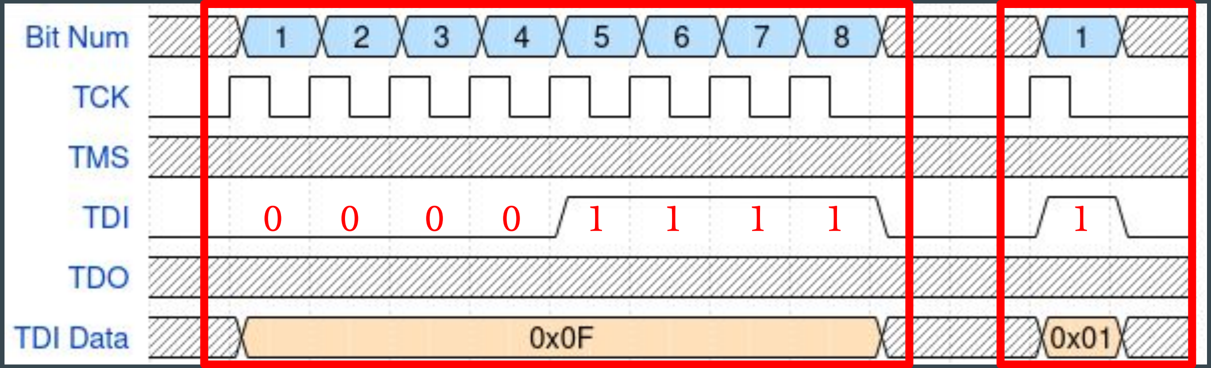
0	0	0	0	1

Boundary Scan IR

1	1	1	1	1

2. Transfer the IDCODE register address to the IR

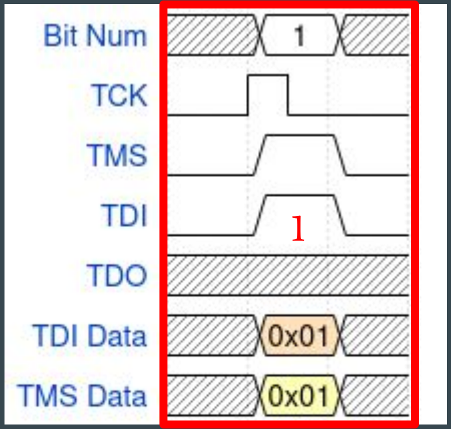
First Two Transfer TDI Cmds



Txfer Byte Cmd

Txfer Bit Cmd

Following Transfer TMS Command



Current IRs

DTM IR					Boundary Scan IR				
0	0	0	0	1	1	1	1	1	1
IDCODE (0b0_0001)					BYPASS (0b1_1111)				

JTAG Select Register Example Steps

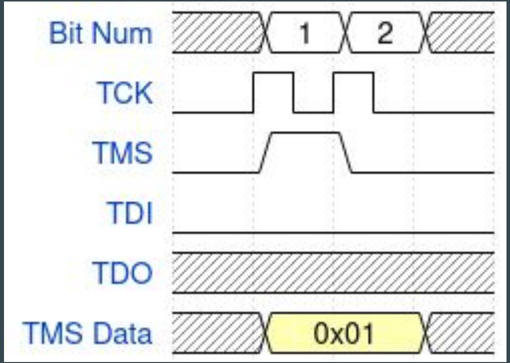
1. Transition DTM TAP from RUN-TEST-IDLE to SHIFT-IR
2. Transfer IDCODE register address to IR
3. Transition DTM TAP back to RUN-TEST-IDLE

3. Transition DTM TAP back to RUN-TEST-IDLE

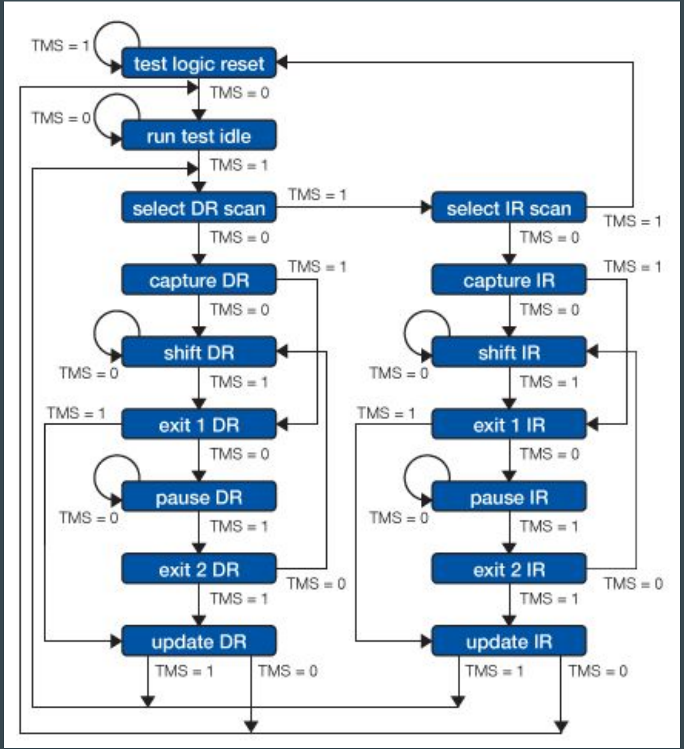
Clock Data to TMS Pin (LSB first, No Read)

4B	02	01
Cmd	Length (bits)	Data (to TMS)

FTDI MPSSE command to transition from EXIT-1-IR to RUN-TEST-IDLE



Timing Diagram for Clk Data to TMS Cmd



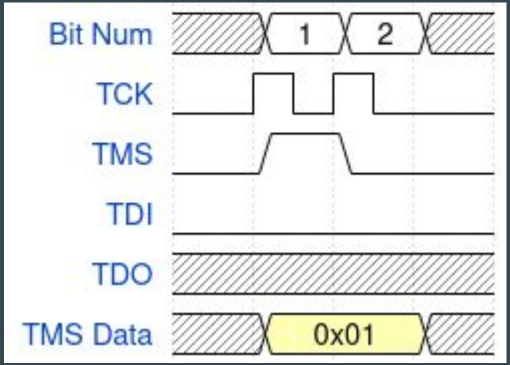
JTAG TAP State Machine

3. Transition DTM TAP back to RUN-TEST-IDLE

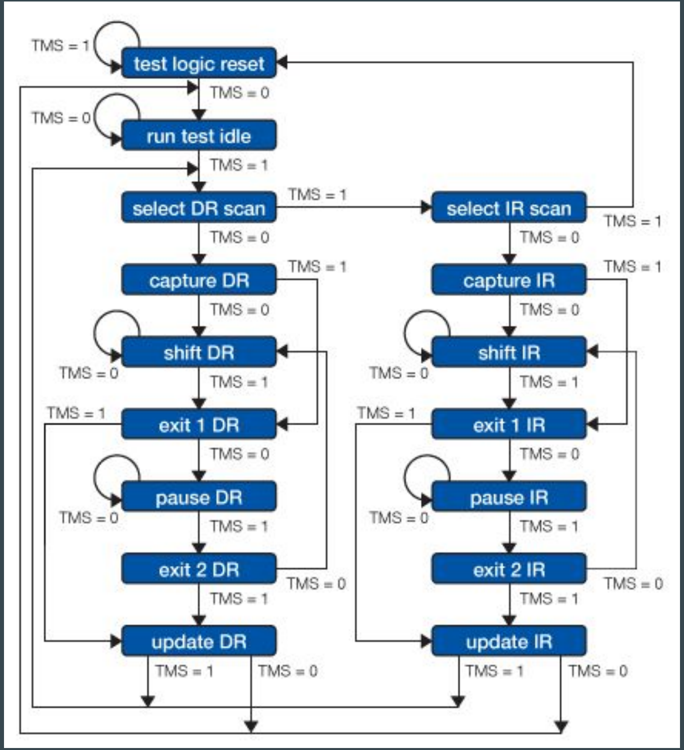
Clock Data to TMS Pin (LSB first, No Read)

4B	02	01
Cmd	Length (bits)	Data (to TMS)

FTDI MPSSE command to transition from EXIT-1-IR to RUN-TEST-IDLE



Timing Diagram for Clk Data to TMS Cmd



JTAG TAP State Machine

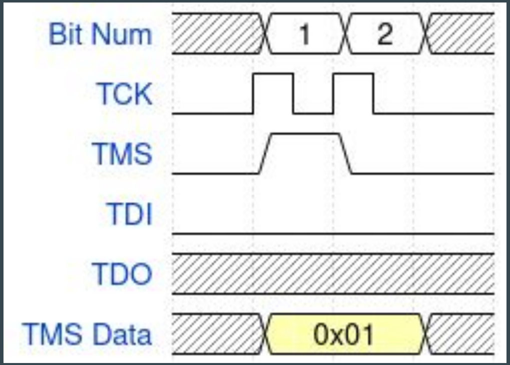
3. Transition DTM TAP back to RUN-TEST-IDLE

Clock Data to TMS Pin (LSB first, No Read)

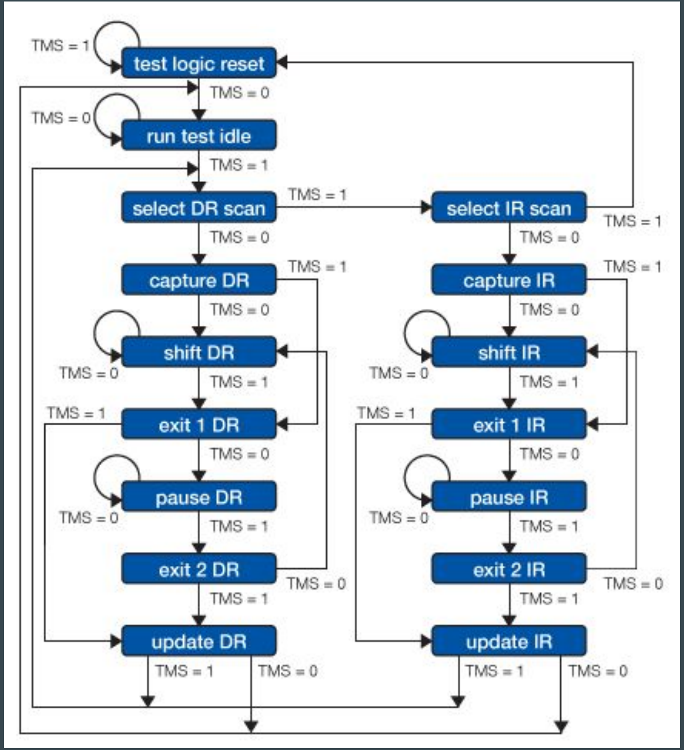
4B	02	01
Cmd	Length (bits)	Data (to TMS)

FTDI MPSSE command to transition from EXIT-1-IR to RUN-TEST-IDLE

(Final cmd in Step 2 transitions from SHIFT-IR to EXIT-1-IR)



Timing Diagram for Clk Data to TMS Cmd



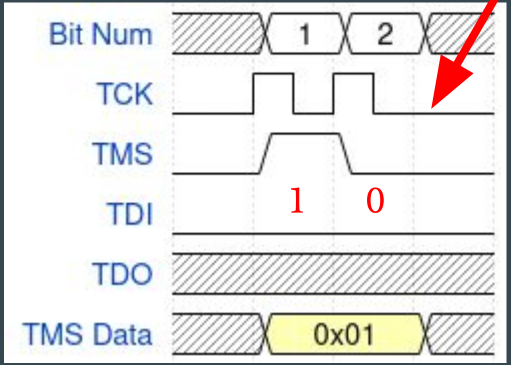
JTAG TAP State Machine

3. Transition DTM TAP back to RUN-TEST-IDLE

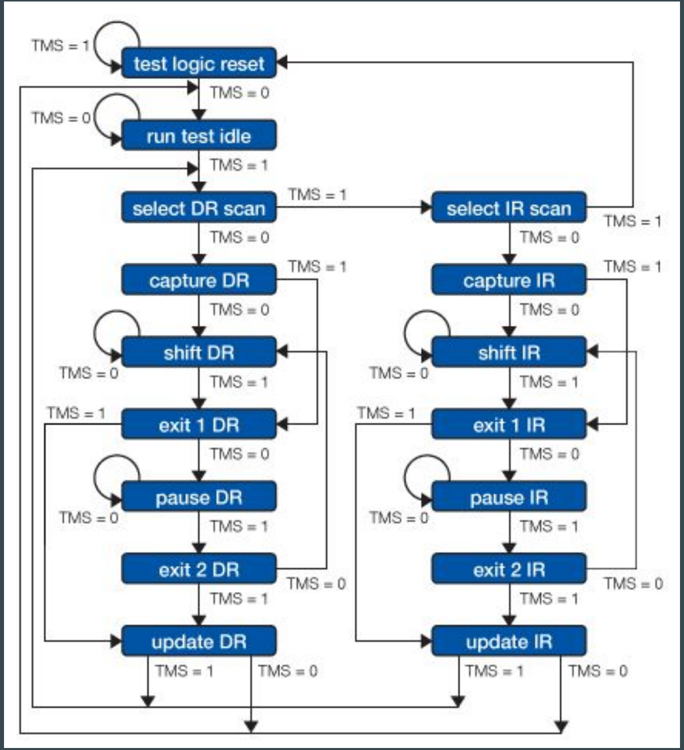
Clock Data to TMS Pin (LSB first, No Read)

4B	02	01
Cmd	Length (bits)	Data (to TMS)

Note reversed ordering again as we use LSB first cmds



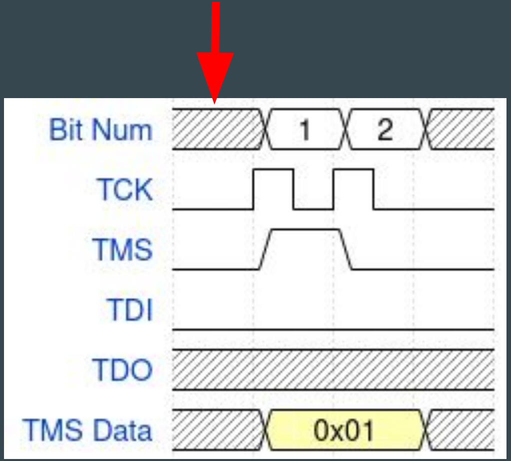
Timing Diagram for Clk Data to TMS Cmd



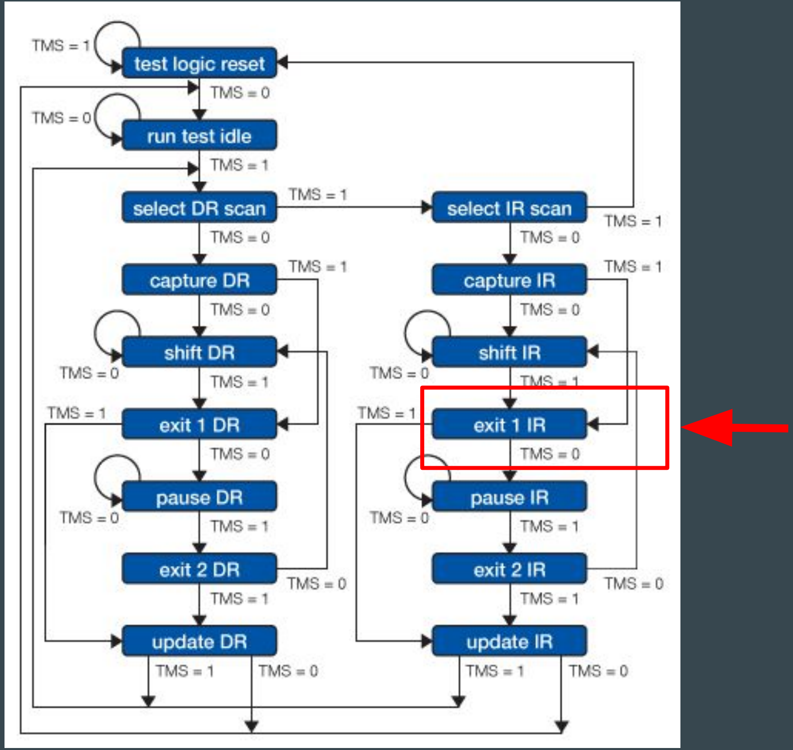
JTAG TAP State Machine

3. Transition DTM TAP back to RUN-TEST-IDLE

Start in EXIT-1-IR



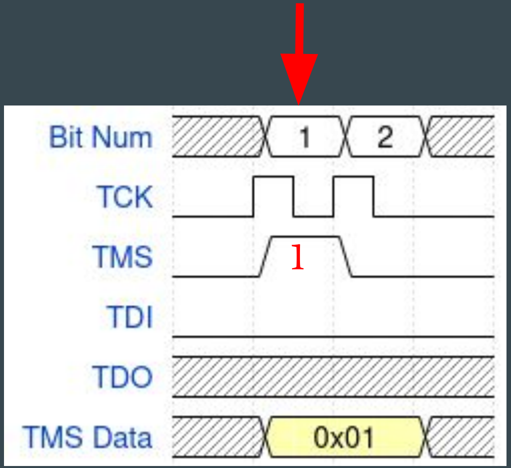
Timing Diagram for Clk Data to TMS Cmd



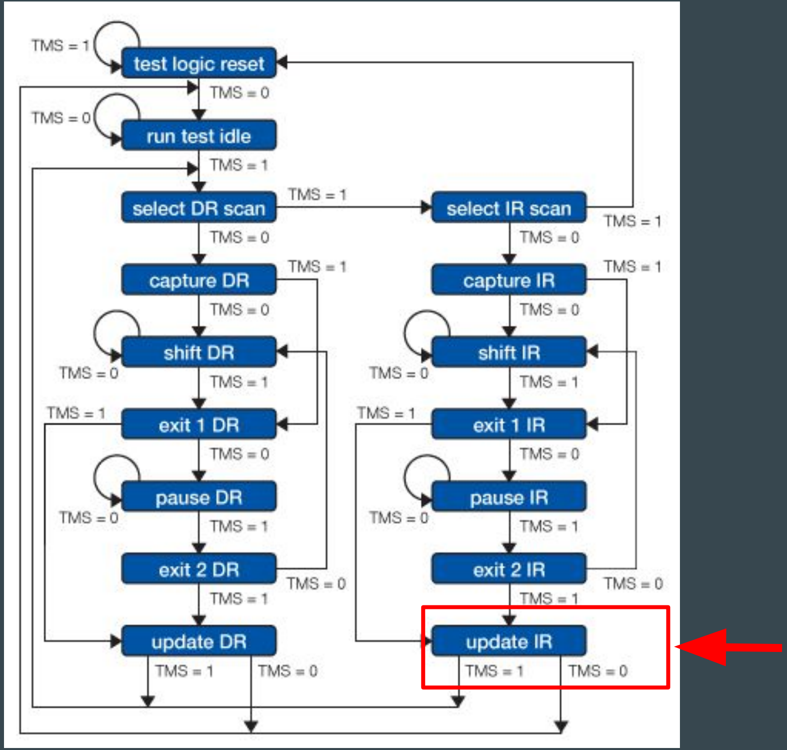
JTAG TAP State Machine

3. Transition DTM TAP back to RUN-TEST-IDLE

TMS bit is 1, go to UPDATE-IR



Timing Diagram for Clk Data to TMS Cmd

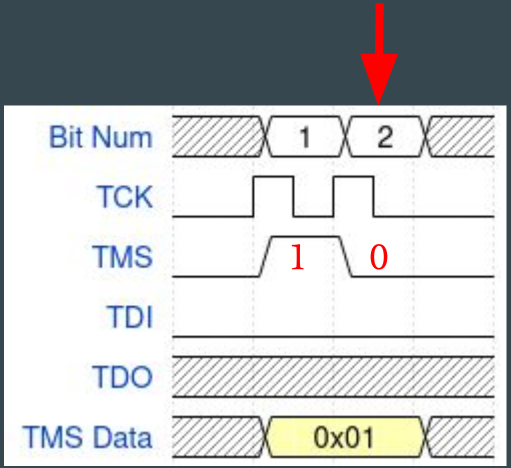


JTAG TAP State Machine

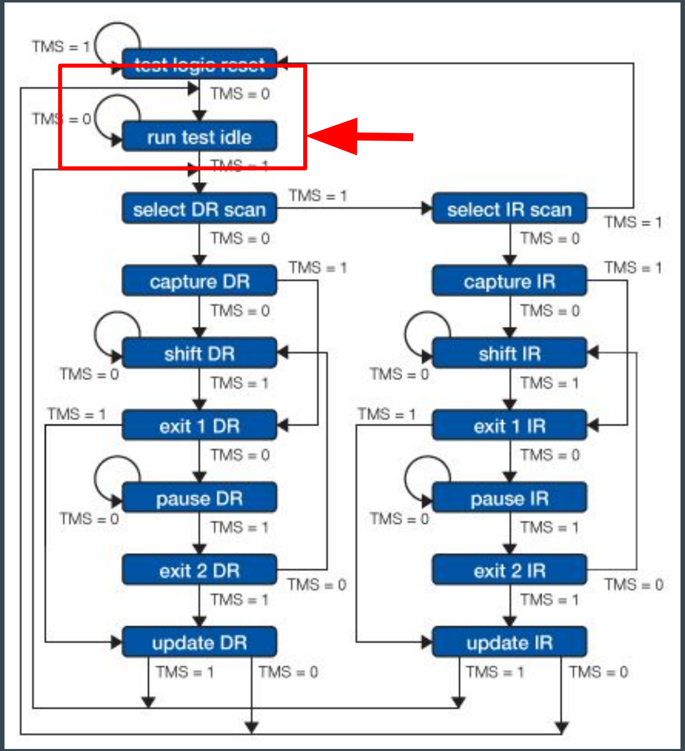
3. Transition DTM TAP back to RUN-TEST-IDLE

TMS bit 0, move to SHIFT-IR

(Can now run next JTAG cmd, likely reading DR to get IDCODE)



Timing Diagram for Clk Data to TMS Cmd



JTAG TAP State Machine

References & Useful Links

- probe-rs Documentation
- FTDI MPSSE Commands Datasheet
- RISC-V Debug Spec v0.13.2
- 1BitSquared Tigard Protocol Tool and BitMagic Basic Logic Analyzer
- Sigrok PulseView and Wireshark
- DEFCON 23 JTAG Hacking Talk
- Programming SRAM w/ SWD (ARM-specific)