NFC Forum Type 2 Tag Platform Operations with the TRF7970A

NFC/RFID Training Module (2014)

S2 MCU NFC/RFID Applications Team

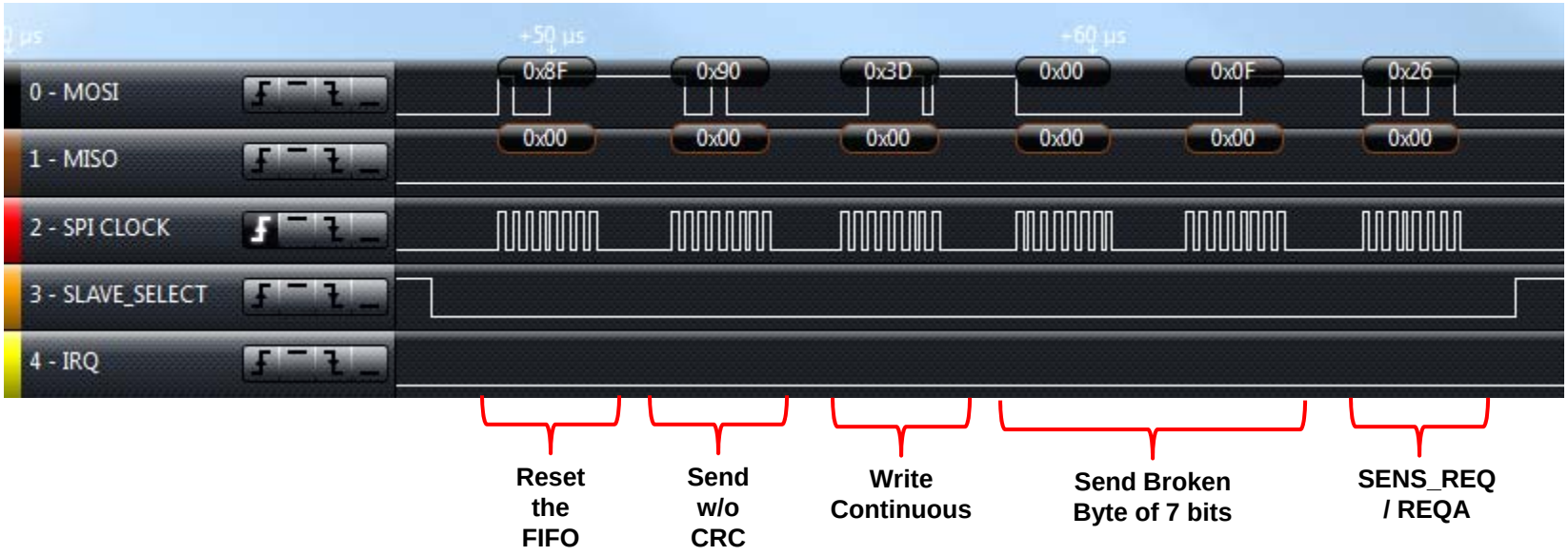
Texas Instruments

Overview of NFC Forum Type 2 Tag Platform Operations

- TRF7970A being used with NFC Forum Type 2 Tag Platform operations is possible using Direct Mode 2 (default mode of the TRF7970A)
- TRF7970A will be configured for ISO14443A operations by MCU
- TRF7970A (+ MCU) will activate and select the Type 2 tag platform using ISO14443A standard command flow
- After activation and selection, detection of the NDEF message will be done using the READ command, starting on Block 3.
  - If tag is not previously NDEF Formatted, then obviously writing to Blocks 3, 4 and 5 would be first step to take with the tag, after activating and selecting it.
- Then the reading or writing of the NDEF Message is possible on the remaining block values.
- Reference ➔ **Type 2 Tag Operation Specification, Technical Specification**
  - T2TOP 1.1, NFC Forum™, NFCForum-TS-Type-2-Tag_1.1 (Dated 2011-05-31)

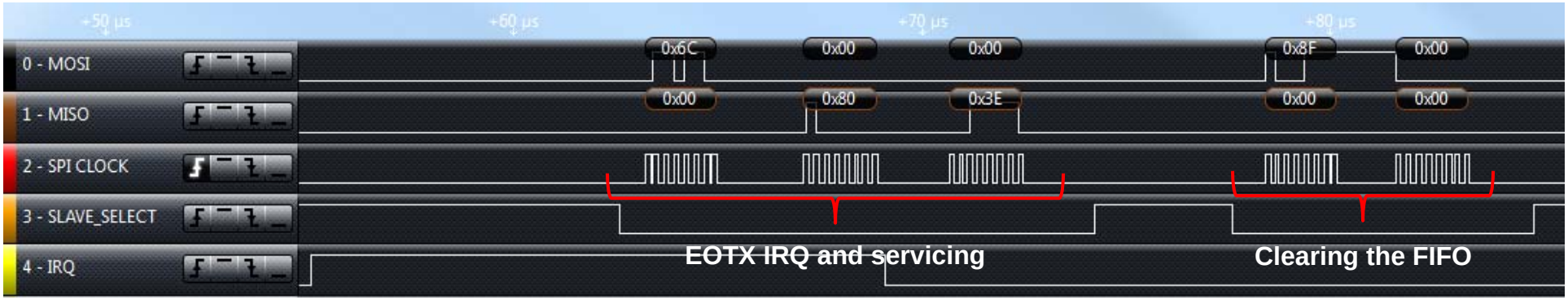
Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- First we configure the TRF7970A as ISO14443A reader/writer by writing registers:
  - 0x09 ➔0x01 (100% MOD depth, for ISO14443A)
  - 0x00 ➔0x21 (+5VDC ops, Full TX power out)
  - 0x01 ➔0x88 (no CRC expected in response)
- Then we issue SENS_REQ (a.k.a REQA) to activate the card:
  - 0x8F, 0x90, 0x3D, 0x00, 0x0F, 0x26, where 0x0F is indicating that a broken byte of seven bits will be sent out and 0x26 is that broken byte called SENS_REQ (NFC Forum name) or REQA (ISO14443A name)

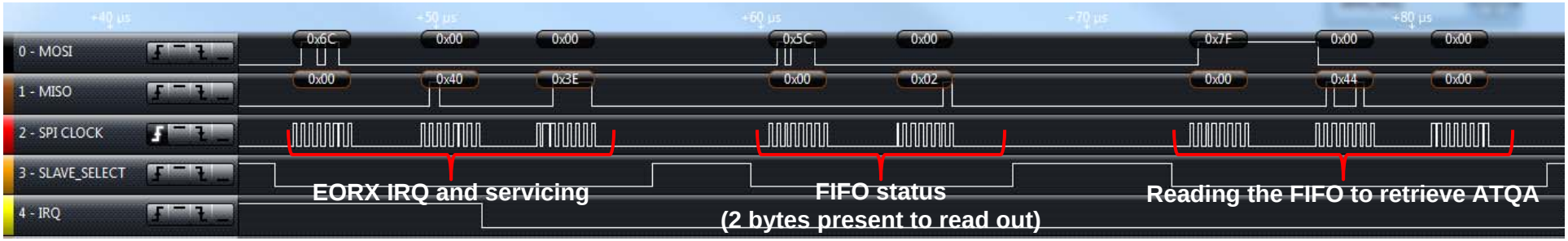


Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- Then, EOTX IRQ is received, serviced, FIFO is cleared.



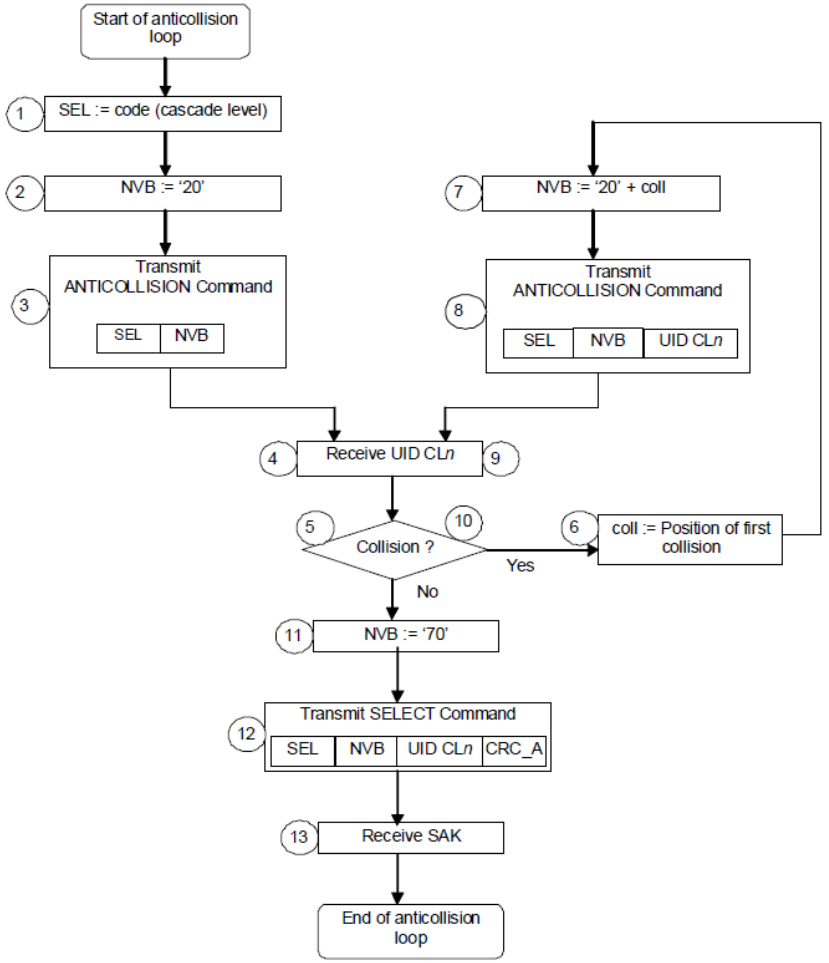
- Next EORX IRQ is received and serviced, FIFO status is read, then FIFO is read out to retrieve the SENS_RES (NFC Forum name) or ATQA (ISO14443A name), **card is now activated**.



Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- To select the Type 2 tag, the ISO14443A anti-collision loop logic is followed ➔➔➔

1. The PCD shall assign SEL with the code for the selected anti-collision cascade level.
2. The PCD shall assign NVB with the value of '20'. NOTE This value defines that the PCD will transmit no part of UID CL*n*. Consequently this command forces all PICCs in the field to respond with their complete UID CL*n*.
3. The PCD shall transmit SEL and NVB.
4. All PICCs in the field shall respond with their complete UID CL*n*.
5. If more than one PICC responds, a collision may occur. If no collision occurs, steps 6 to 10 shall be skipped.
6. The PCD shall recognize the position of the first collision.
7. The PCD shall assign NVB with a value that specifies the number of valid bits of UID CL*n*. The valid bits shall be part of the UID CL*n* that was received before a collision occurred followed by a (0)b or (1)b, decided by the PCD. A typical implementation adds a (1)b.
8. The PCD shall transmit SEL and NVB, followed by the valid bits.
9. Only PICCs of which the part of UID CL*n* is equal to the valid bits transmitted by the PCD shall transmit their remaining bits of the UID CL*n*.
10. If further collisions occur, steps 6 to 9 shall be repeated. The maximum number of loops is 32.
11. If no further collision occurs, the PCD shall assign NVB with the value of '70'. NOTE This value defines that the PCD will transmit the complete UID CL*n*.
12. The PCD shall transmit SEL and NVB, followed by all 40 bits of UID CL*n*, followed by CRC_A.
13. The PICCs which UID CL*n* matches the 40 bits shall respond with their SAK.
14. If the UID is complete, the PICC shall transmit SAK with cleared cascade bit and shall transit from READY state to ACTIVE state or from READY* state to ACTIVE* state.
15. The PCD shall check if the cascade bit of SAK is set to decide whether further anti-collision loops with increased cascade level shall follow.

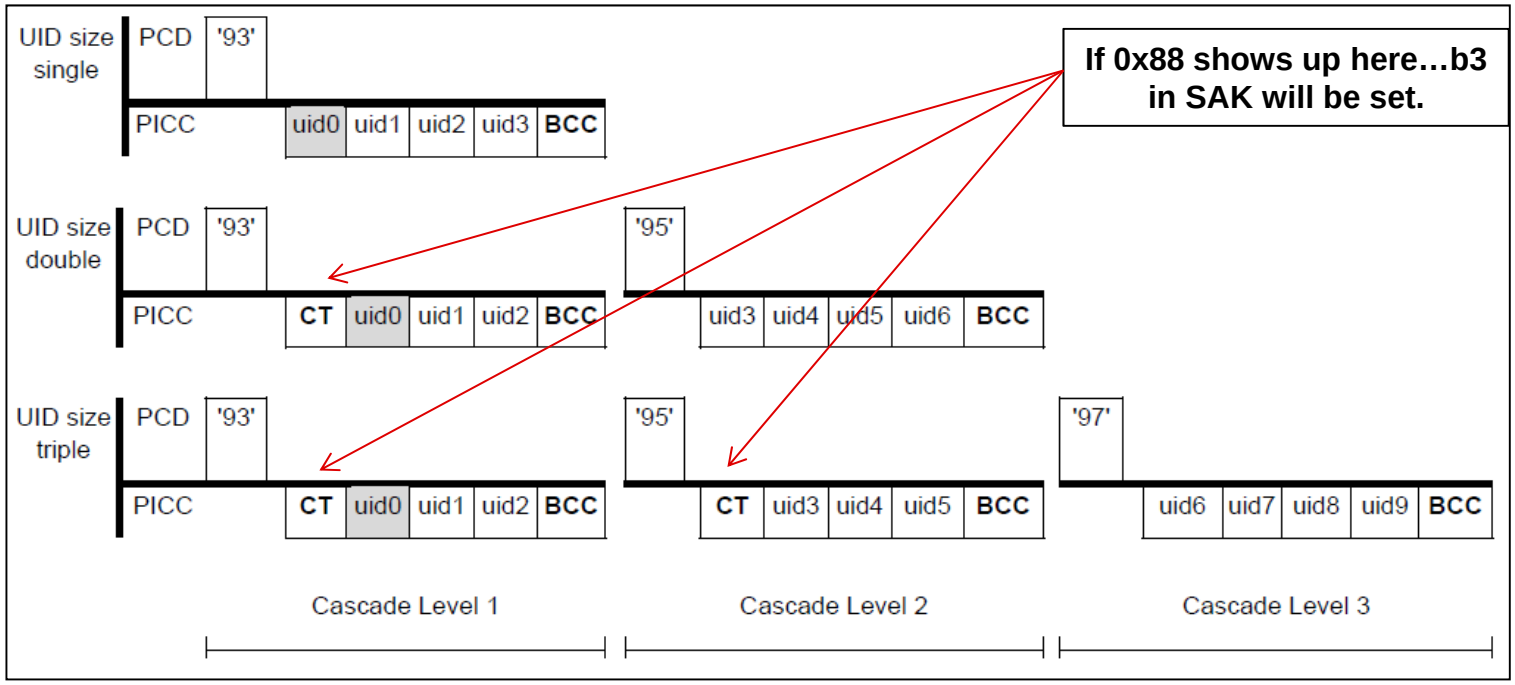


Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- The following algorithm shall apply to the PCD to get the complete UID:

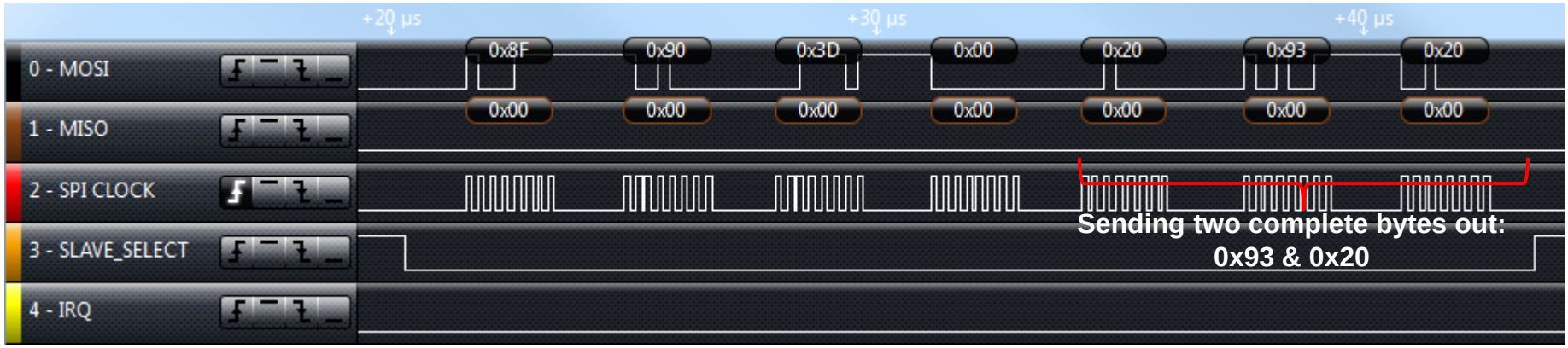
1. The PCD selects cascade level 1.
2. The anti-collision loop shall be performed.
3. The PCD shall check the cascade bit of SAK.
4. If the cascade bit is set, the PCD shall increase the cascade level and initiate a new anti-collision loop.

- **NOTE: a 0x88 showing up in the UID0 and UID3 spot (shown as CT for double and triple size UIDs) is a pointer or clue that UID will not be complete in that cascade round. 0x88 is not allowed for UID3 in double size UID tags**



Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- After activation, card must be selected. This is started by sending out the first anti-collision command.



- The EOTX IRQ is received and serviced, along with the EORX IRQ is received, FIFO status is read, then FIFO is read out to retrieve the UID_CL*n.*

Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

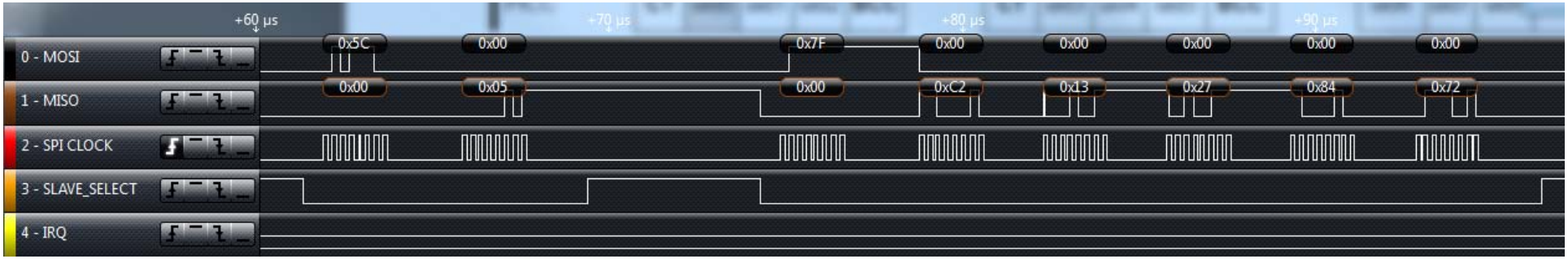
- The card must be selected before looping back for next cascade level. This is done by sending out the SEL + NVB + UIDCLn + CRC_A (done by the TRF79xxA, because 0x91 sent out in the command string.
  - NOTE: ISO Control register must be changed to 0x08 at this time because the CRC will be coming back from the tag.)

- The EOTX IRQ is received and serviced, along with the EORX IRQ is received, FIFO status is read, then FIFO is read out to retrieve the SAK.
- Here we can see that b3 in the SAK response is set, also indicating UID is not complete and according to the rules, the cascade level must be incremented and anti-collision loop performed again.

Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- The cascade level is incremented and the second anti-collision command is sent out. (0x95, 0x20)

- The EOTX IRQ is received and serviced, along with the EORX IRQ is received, FIFO status is read, then FIFO is read out to retrieve the UID_CL*n.*



Activating & Selecting an unformatted Type 2 tag (Mifare Ultralight, Ultralight C or Infineon my-d™ move / my-d™ NFC)

- The card must be selected. This is done by sending out the SEL + NVB + UIDCLn + CRC_A (done by the TRF79xxA, because 0x91 sent out in the command string.
  - NOTE: ISO Control register must be changed to 0x08 at this time because the CRC will be coming back from the tag.)

- The EOTX IRQ is received and serviced, along with the EORX IRQ is received, FIFO status is read, then FIFO is read out to retrieve the SAK.
- Here we can see that b3 in the SAK response is not set, indicating the UID is now complete and b6 not being set indicates that the device is not ISO14443-3 compliant (which is correct for Type 2 tag).
- The MCU can concatenate the response to form the complete double size UID
  - In this case it is: **04:E8:1C:C2:13:27:84**

READ Command for Tag Type 2

- The READ command has command code 0x30 and needs the block number (Bno) as a parameter.
- The Type 2 Tag Platform responds to a READ command by sending 16 bytes, starting from the block number defined in the READ command
  - (for example: if BNo is equal to 0x03, then blocks 3, 4, 5, and 6 are returned).
- The block numbering and memory organization of the Type 2 Tag Platform is described in previous slide.
- In case of error, the Type 2 Tag Platform sends a NACK response.

Length Byte Number		**1 byte** **Byte 1**	**1 byte** **Byte 2**
Command	READ Command	Code: 30h	BNo: 00h - FFh
Length Byte Number		**16 bytes** **Byte 1 – 16**	
Response	READ Response	16 bytes payload	-
	NACK Response	See Section 5.4 (4 bits)	-

(from NFC Forum TT2 OS, section 5.1)

NFC Forum Type 2 Tag Platform Memory Maps

- First Three Blocks of these tags contain the UID, etc. (as listed)
- Block 3 is the capability container, Block(s) 4 (& 5) contain the TLV information, depending on the memory structure.
- Subsequent blocks are the NDEF message area

Byte Number	0	1	2	3	Block
UID / Internal	Internal0	Internal1	Internal2	Internal3	0
Serial Number	Internal4	Internal5	Internal6	Internal7	1
Internal / Lock	Internal8	Internal9	Lock0 = 00h	Lock1 = 00h	2
CC	CC0 = E1h	CC1 = 10h	CC2 = 06h	CC3 = 00h	3
Data	NDEFMessage TLV0 = 03h	NDEFMessage TLV1 = 00h	TerminatorTLV0 = FEh	Data3	4
Data	Data4	Data5	Data6	Data7	5
Data	Data8	Data9	Data10	Data11	6
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	..	Data46	Data47	15

Static Memory Structure

This memory structure is used by Type 2 Tag Platform with a physical memory size equal to 64 bytes.

Byte Number	0	1	2	3	Block
UID / Internal	Internal0	Internal1	Internal2	Internal3	0
Serial Number	Internal4	Internal5	Internal6	Internal7	1
Internal / Lock	Internal8	Internal9	Lock0 = 00h	Lock1 = 00h	2
CC	CC0 = E1h	CC1 = 10h	CC2 = 0Ch	CC3 = 00h	3
Lock Control TLV	LockControlTLV0 = 01h	LockControlTLV1 = 03h	LockControlTLV2 = E0h	LockControlTLV3 = 06h	4
Lock Control TLV / Memory Control TLV	LockControlTLV4 = 33h	MemoryControl TLV0 = 02h	MemoryControl TLV1 = 03h	MemoryControl TLV2 = E1h	5
Memory Control TLV / NDEF Message TLV	MemoryControl TLV3 = 0Fh	MemoryControl TLV4 = 03h	NDEFMessage TLV0 = 03h	NDEFMessage TLV1 = 00h	6
Terminator TLV / Data	TerminatorTLV0 = FEh	Data13	Data14	..	7
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	..	..	..	.
Data	..	Data93	Data94	Data95	27
Lock / Reserved	Lock2 = 00h	Reserved0	Reserved1	Reserved2	28
Reserved	Reserved3	Reserved4	Reserved5	Reserved6	29
Reserved	Reserved7	Reserved8	Reserved9	Reserved10	30
Reserved	Reserved11	Reserved12	Reserved13	Reserved14	31

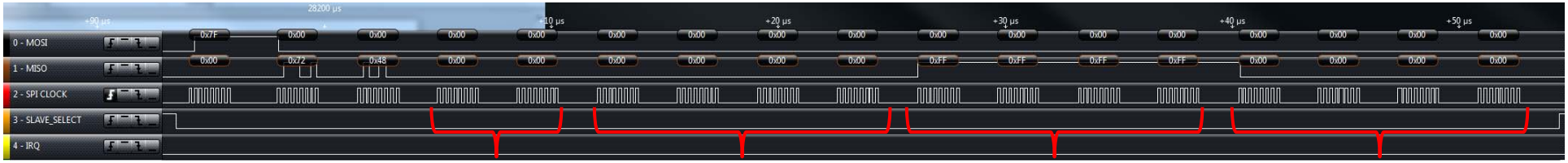
Dynamic Memory Structure

This memory structure is applied to Type 2 Tag Platform with a memory size bigger than 64 bytes.

Reading Data from NFC Forum Type 2 Tag Platform (example is empty / non-formatted card)

- Next step in the process, now that the tag is selected, is to read the blocks for the lock bits and the Capability Container.
- The NFC Forum Tag Type 2 command for this (as shown earlier in slide 4) is the READ command along with the starting block number.
- For this we can send out the command string 0x8F, 0x91, 0x3D, 0x00, 0x30, 0x02

- The IRQs are received and serviced as before, and FIFO status is read which indicates 16 bytes are present to read as the response from the tag. (four blocks)



Lock Status (unlocked) | **Block 3 Capability Container (not NDEF formatted)** | **Block 4** | **Block 5**

WRITE Command for Tag Type 2

- The WRITE command has the command code A2h followed by the block number (BNo) parameter.
- The NFC Forum Device SHALL use the WRITE command for programming data. This MAY be the CC bytes the lock bytes, or the data area bytes.
- The NFC Forum Device SHALL use the WRITE command block-wise, programming 4 bytes at once.
- If the WRITE command is executed successfully by the Type 2 Tag Platform, the ACK response is sent back.
- In case of error, the Type 2 Tag Platform sends a NACK response.

Table 4: WRITE

Length Byte Number		**1 byte** **Byte 1**	**1 byte** **Byte 2**	**4 bytes** **Byte 3 – 6**
Command	WRITE Command	Code: A2h	BNo: 00h – FFh	Data
Length Byte Number		**4 bits** **-**		
Response	ACK Response	See Section 5.4	-	-
	NACK Response	See Section 5.4	-	-

(from NFC Forum TT2 OS, section 5.2)

Writing Data / Formatting
NFC Forum Type 2 Tag Platform

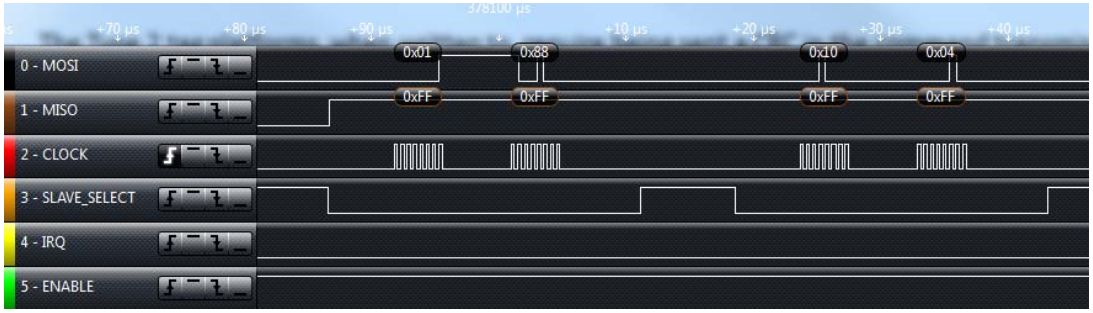
- Its very important to get this part correct as it pertains to Block 3 on Type 2 tags as this is usually an One time programmable (OTP) section (i.e. Mifare Ultralight, Ultralight C)
- Blocks 3 & 4 (and 5, if dynamic memory structure tag) need to be written to in order for the tag to then be considered NDEF Formatted.
- **Once again**, Block 3 is **OTP**, so make sure you are writing exactly what should be there beforehand!

Writing Data / Formatting
NFC Forum Type 2 Tag Platform

- The Type 2 tag platforms, when written to, require being sent a CRC in the command transmission, but they respond with a 4 bit ACK or NACK and **no** CRC.
- This means that when using the TRF7970A, Type 2 platform writing (done after the SAK has been received) requires the ISO Control register (0x01) to be written to with a 0x88 (to receive with no CRC) and the Special Functions register (0x10) to be written with a 0x04 (setting bit 2), which enables the 4 bit RX.
- Here we see this technique on the next slides.

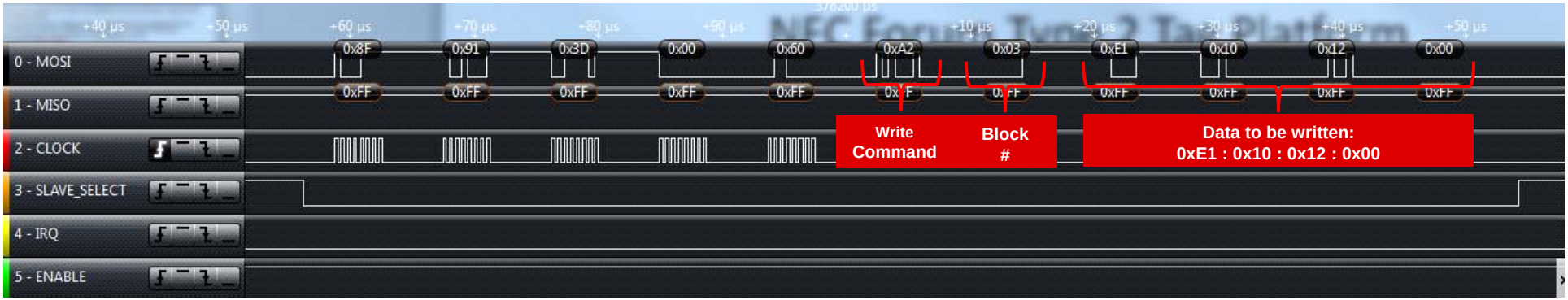
Writing Data / Formatting NFC Forum Type 2 Tag Platform

- First image is writing to register 0x01 with a 0x88 (for ISO14443A @ 106kpbs and receiving the tag response with no CRC, then writing to register 0x10 with a 0x04, which enables the 4 bit receive that we are expecting back from the tag to indicate an ACK or NACK on the write operation.



- Second image is showing the write operation to the capability container area of this tag type (Block 3), which uses: 0xA2 (write command) + the block # (here is 0x03) and the data to write (which is: **0xE1 : 0x10 : 0x12 : 0x00**)

From section 6.1 in T2T Op Spec: 0xE1 = "magic number", 0x10 = version #, 0x12 = 144 bytes memory size of this tag, 0x00 = open R/W access



Writing Data / Formatting NFC Forum Type 2 Tag Platform

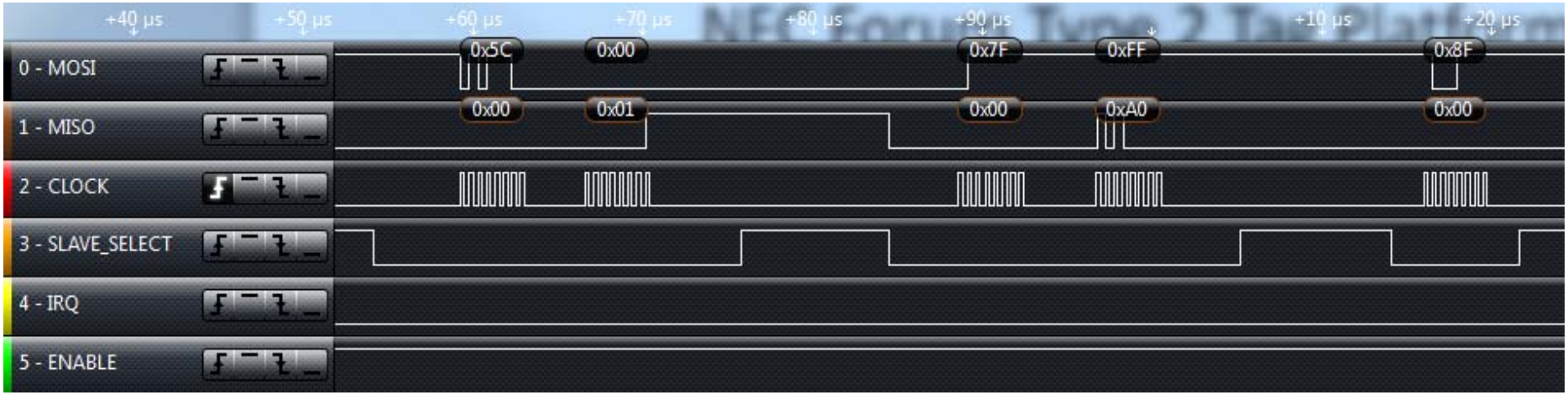
- This results in the EOTX interrupt coming in and then being handled, then EORX IRQ coming in and being handled, with FIFO status being 0x01 and the byte returned for ACK is 0xA0, the upper nibble of course being the 4 bit RX that we expect. (a NACK is returned in same format)
  - From Tables 53 & 55 (NACK and ACK Response Formats) in the Type 2 Tag Platform section of the NFC Forum Digital Protocol Specification)

Table 53: NACK Response Format

b4	b3	b2	b1	Meaning
0	x	0	x	The NACK Response has a value of 0h, 1h, 4h, or 5h.

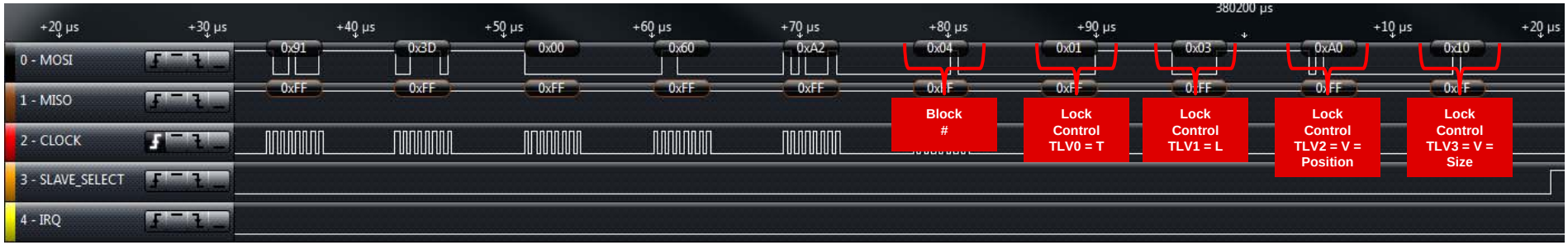
Table 55: ACK Response Format

b4	b3	b2	b1	Meaning
1	0	1	0	The ACK Response has a value of Ah.



Formatting NFC Forum Type 2 Tag Platform

- To write subsequent blocks, you would then increment the Bno and add in the data you desire (next block(s) would be TLV values and then NDEF data content.
- Below are examples of writing Blocks 4 and 5 on a Dynamic Memory structure tag for the NDEF Message TLV
- This is example of NDEF Formatted, but empty NDEF Type 2 tag platform



Writing NDEF Data to NFC Forum Type 2 Tag Platform

- To write subsequent blocks, you would then increment the Bno and add in the data you desire (next block(s) would be TLV values and then NDEF data content.
- Below are examples of writing Blocks 5 and x on a Dynamic Memory structure tag for the NDEF Message TLV and the NDEF Message itself of, in this case: Hello, World!

Writing NDEF Data to NFC Forum Type 2 Tag Platform

- To write subsequent blocks, you would then increment the Bno and add in the data you desire (next block(s) would be TLV values and then NDEF data content.
- Below are examples of writing Blocks 5 and x on a Dynamic Memory structure tag for the NDEF Message TLV and the NDEF Message itself of, in this case: Hello, World!