AN4433
Application note

Storing data into the NDEF memory of M24SR

Introduction

M24SR series Dynamic NFC Tags is a series of dynamic NFC Forum Type 4 Tags, based on ISO/IEC 14443 Type A technology:

- Dynamic tag: EEPROM area can be accessed either from I^2C or RF interface.
- NFC Forum Type 4 Tag: memory organization and access complies with the associated NFC Forum specification.

The purpose of this application note is to give some guidelines on how to access and organize the memory of M24SR products, as per the NFC Forum standards and M24SR specificities.

This document specifically proposes some mechanisms to store proprietary data, with or without standard NFC data (NDEF).

Contents

List of tables

List of figures

1 Reminder on NFC Forum Type 4 Tag and NDEF data structure

1.1 Memory organization

Memory of an NFC Forum Type 4 Tag is organized in several files:

- 1 “Capability Container” (CC) file: this read-only file describes the other files that are present in the memory, by consecutive description blocks (called TLV blocks, 1 TLV block per file)
- 1 mandatory NDEF file: this file is necessarily the first one described in CC file
- 0, 1 or more proprietary files: if any, proprietary files are described in following TLV blocks
- A TLV block gives information on a file: file ID, file length, read and write access rights.

See reference document number *2.* for more information on NFC Forum Type 4 Tag files organization.

In addition to these files, M24SR products contain a proprietary System file, to handle M24SR specific advanced parameters/features (see reference document number *6.*).

An NDEF file contains the NDEF message length in the first 2 bytes, and an NDEF message of 1 or more NDEF records.

Each record contains a consistent set of data, of a specific type (Text, URI, MIME type…)

These data types are what we call “standard NDEF data”.

Table 1. Example of an NDEF message with a set of records

NDEF Message						
R_1 MB=1	...	R_r	...	R_s	...	R_t MB=1

A set of data can be split into several records, called “chunk” records.

The length of a record can be deduced by the interpretation of its header: this implies that one needs to parse the headers of the “n-1” records, to get record number “n”.

For more information on NDEF message format, see reference document number *3.*

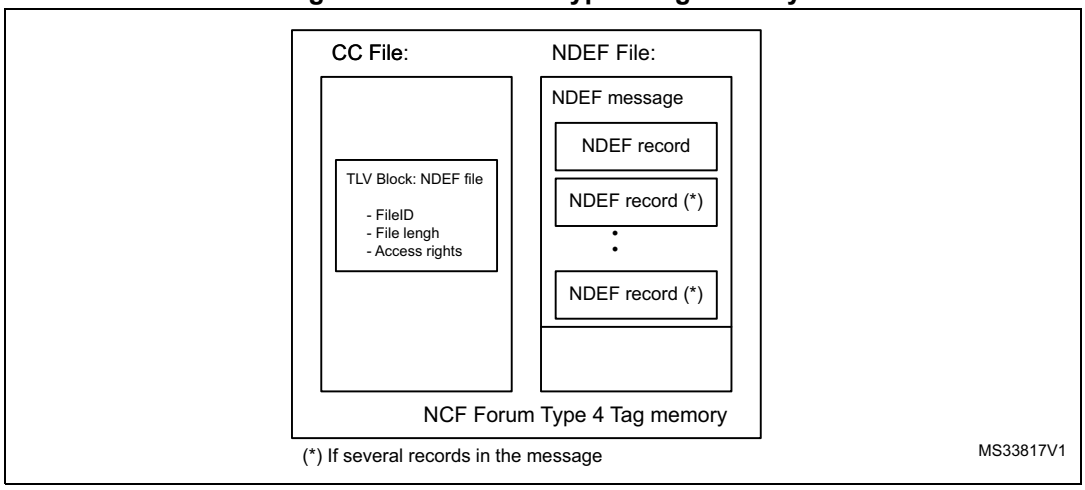
A Proprietary file contains the length of the data on the 2 first bytes, then proprietary data (see reference document number *2.*).

In this document, “proprietary data” refer to any data set, not standardized by NFC Forum in NDEF format.

These data can be, for example:

- User settings
- User data
- Calibration data
- Event logs
- System information
- …

Figure 1. NFC Forum Type 4 Tag memory



1.2 Memory access

An NFC Forum Type 4 Tag can be accessed only through a “file access” interface: no direct addressing of memory area is possible.

File access allows to:

- have direct access to a specific area of the memory, without parsing the other part
- separate the memory areas for different applications/data (with no accidental overread/overwrite of the memory)
- give read/write access rights per memory area

To interact with an NFC Forum Type 4 Tag, a reader/writer device must:

- apply the “Listen, RF Collision Avoidance, Technology Detection, Collision Resolution” activity for suitable technology (NFC-A technology for M24SR products), as defined in reference document number *1.*
- perform the “Device Activation” activity for suitable Device Platform (Type 4A Tag Platform for M24SR products), as defined in reference document number *1.*
- apply the “NDEF Detection Procedure” (and Proprietary files read, if any), as defined in reference document number *2.*, inside ISODEP protocol (see reference document number *1.*).

For examples of file accesses:

- See *Appendix A* and *Appendix B*.

1.3 Reader/Writer behavior in front of NFC Forum Type 4 Tag

NFC is now widely deployed on both common platforms (smartphones and tablets) with well-known OS (Android™, Windows® Phone 7.5/8.0, BlackBerry® 10), and proprietary NFC/RFID readers.

NFC Forum Type 4 Tag and NDEF data format offer native interoperability to exchange standard NDEF data between these devices.

In front of an NFC Forum Type 4 Tag, any reader will natively interpret the standard NDEF data, and start the application, registered for the data type of the first record of the NDEF file.

If no application is registered for the data type, default behavior will be executed:

- launch web browser for an URL.
- route audio from loudspeaker to distant box, in case of BT pairing to audio capable module.
- initiate a phone call, if telephone number is stored.
- ….

2 M24SR specificities

2.1 Memory access through I²C

An NFC Forum Type 4 Tag can be read through RF interface, by applying procedure described in *Section 1.2*

M24SR products offer an I²C interface, that gives the same “file access” possibility: an application has to apply the “NDEF Detection Procedure”, inside ISODEP protocol, on top of I^2C protocol.

This serial interface makes the tag “dynamic”, as its content can be changed by the application, all along the tag lifetime.

- See reference document number *2.* for “NDEF Detection Procedure”.
- See reference document number *1.* for ISODEP protocol.
- See reference document number *6.* for I^2C protocol.

For examples of file accesses through I^2C.

- See *Appendix C*.

2.2 ExtendedReadBinary command

Basically, when reading data in an NDEF file of an NFC Forum Type 4 Tag, the ReadBinary command only allows reading the NDEF message of the file.

The Write command (UpdateBinary) allows writing data on the whole file.

See *Section 2.4* for more visibility on NDEF file / NDEF message / memory beyond NDEF message.

See reference document number *2.* for these procedures.

But this lets unused memory area at the end of the NDEF file.

For example:

- M24SR64-Y: 8190 available bytes for NDEF message
- Message content: URL “http://www.st.com/nfc-rfid”
- NDEF message length (1 NDEF record): 20 bytes
- 8170 remaining non-readable bytes in NDEF file

Table 2 shows the NDEF file layout, where the non-readable bytes are highlighted in gray.

Table 2. NDEF file layout

Address	Data															
0x0000	0x00 (1)	0x14 (1)	0xD1 (2)	0x01 (2)	0x10 (2)	0x55 (2)	0x01 (2)	0x73 (2)	0x74 (2)	0x2E (2)	0x63 (2)	0x6F (2)	0x6D (2)	0x2F (2)	0x6E (2)	0x66 (2)
0x0010	0x63 (2)	0x2D (2)	0x72 (2)	0x66 (2)	0x69 (2)	0x64 (2)	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0x07F0	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--	0x--

1. NDEF Message length.
2. NDEF message content.

To be able to read beyond NDEF message, M24SR products offer a proprietary command: ***ExtendedReadBinary*** command.

This command can read the bytes *beyond NDEF message*, until the end of the file.

See reference document number *6.* for more information on this command.

2.3 UpdateFileType command

NFC Forum Type 4 Tag has 2 types of data files: NDEF and Proprietary files.

Natively, M24SR products contain 1 mandatory NDEF file.

M24SR products offer a proprietary command to turn the type of the file from one type to the other ***UpdatedFiletype***

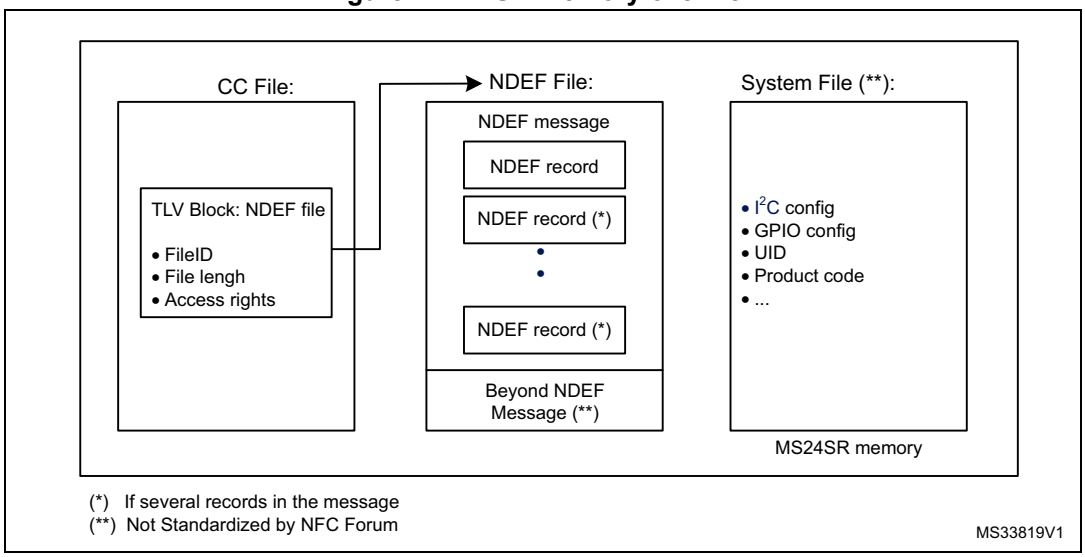
For more details on this command, see reference document number *6.*

Warning:

– Turning NDEF file to Proprietary one, makes the tag no more NFC compliant; only proprietary application will be able to deal with the tag (see *Section 1.1*).

2.4 M24SR memory overview

Figure 2. M24SR memory overview



Warning:

- M24SR memory cannot be accessed (read and write) through RF interface, if an I^2C session is opened.
- M24SR memory cannot be accessed (read and write) by default through I^2C interface, if an RF session is opened.
- I^2C interface benefits from an advanced command (KillRFSession), to force the end of the RF session, and open its own one.
- For more details about M24SR specificities see *6.*

3 Storing proprietary data in M24SR

3.1 Beyond NDEF message

As seen in *Section 2*, memory area beyond NDEF message can be used to store data, apart from standard NDEF data.

A Proprietary application using ST Proprietary *ExtendedReadMemory* command, can access this area (read and write) for its own purpose.

Warning:

- A non-proprietary application will not be aware about the use of this area.
- If it has write access rights to the NDEF file, an application may corrupt the proprietary data when writing an NDEF message.

3.2 Proprietary NDEF record

When defining NDEF format, NFC Forum thought about providing a way to store proprietary data in a specific record of an NDEF message.

This is possible by using a record with:

- TNF = “NFC Forum external type” (0x04)
- Type refers to a proprietary domain, and associated application

For more details about the way to construct such a record, see reference documents number *2.* and *4.*

A commonly used example of a proprietary NDEF record, is the Android Application Record (AAR): this record is intended to Android system, and stores the name of the preferred application to execute, when this tag is read by an Android device.

An AAR has:

- TNF = 0x04 (“NFC Forum external type”)
- Type = “android.com:pkg”
- Payload = package name of the application

For more information on AAR, refer to Android documentation.

See *Appendix B* for an example of NDEF proprietary record, in an NFC Forum Type 4 Tag.

3.3 Proprietary file

With the help of ST proprietary UpdateFileType command, a M24SR memory administrator can turn the M24SR file into a Proprietary one (see *Section 2.3*)

Advantages:

- Tag can be fully dedicated to proprietary data with immediate access to the data, from the beginning of the file

Drawbacks:

- Not NFC compliant: 1st file of the tag must be an NDEF one (see *Section 1.1*)
- Needs a proprietary application to interact with it: proprietary reader or proprietary application on Android device (needed Android device should be fully chosen and validated, as some may not recognize the tag, depending on NFC chipset and stack)

4 Combining standard NDEF and proprietary data in M24SR

4.1 Using an NDEF file and bytes beyond NDEF message

Configuration:

- Standard NDEF data in NDEF message
- Proprietary data in bytes “beyond NDEF message” see *Section 3.2*

Advantages:

- NFC compliant
- Supported by all common platforms (Android, WP)
- “Fast” accessibility (read/write) to proprietary data, by memory addressing in NDEF file

Drawbacks:

- No isolation of the read/write access rights per data type
- Proprietary data accessible only if the platform API allows proprietary commands (not possible on Windows® Phone 8)
- If third-party application has write access rights to NDEF file, proprietary data may be corrupted.

Tip:

- **It is recommended** to use this configuration, when **NDEF file is set in Read only** for third-party application
- A proprietary application may have suitable access rights to change the content of the NDEF file (known write password).

4.2 Using an NDEF file with a proprietary NDEF record

Configuration:

- Standard NDEF data in NDEF record
- Proprietary data in NDEF proprietary record (see *Section 3.2*)

Advantages:

- NFC compliant: NDEF data structure is still as specified by NFC Forum (see *Section 1.1*)
- Standard NDEF data can natively be read by all common platforms (Android™, WP)
- Proprietary data can be read and written by a proprietary application on all common platforms (Android™, WP)

Drawbacks:

- Accessibility to last record(s) may be slow, as there needs to parse all headers of previous records, to identify the position of the expected one
- No isolation of the read/write access rights per data type
- An application shall take care of all records of the message, if dealing with only one of these (as said in reference document number *3.*)

Tip:

- In case of several NDEF records in the NDEF message, an “association by containment” may be worth, as defined in reference document number *4.*
- This can be realized, for example, through a “SmartPoster” record, as specified in reference document number *5.* Length of this “SmartPoster” record will immediately give the position of following record(s)

5 Decision matrix

Hereafter is a matrix that suggests a solution, according to the configuration and environment of the targeted application.

Remarks:

- “standard” application refers to default behavior of the targeted platform (tag detection and default read), and third-parties softwares (NDEF data read/write)
- “proprietary” application refers to software developed for a specific product with M24SR (for standard NDEF and proprietary data read/write)

Table 3. Decision Matrix

Application environment				Memory configuration		
NDEF data	**Proprietary data**	**Reader/Writer**	**Application**	**1 NDEF file**	**1 proprietary file**	**1 NDEF file + beyond NDEF message**
Yes in Read/Write	No	RF or I^2C proprietary	Proprietary	C	C	C
		Android device	Standard	C	N/A	C
			Proprietary	C	N/A	C
		WP8 device	Standard	C	N/A	C
			Proprietary	C	N/A	C
Yes in Read/Write	Yes	RF or I^2C proprietary	Proprietary	C(1)	C	C
		Android device	Standard	C(1)	N/A	CSR(2)
			Proprietary	C(1)	N/A	C
		WP8 device	Standard	C(1)	N/A	CSR(2)
			Proprietary	C(1)	N/A	CSR(2) (3)
				C(1)	N/A	
Yes in Read Only	Yes	RF or I^2C proprietary	Proprietary	C(1)	C	C
		Android device	Standard	C(1)	N/A	C
			Proprietary	C(1)	N/A	C(4)
		WP8 device	Standard	C(1)	N/A	C
			Proprietary	C(1)	N/A	CSR(3)
No (NDEF area in Read Only)	Yes	RF or I^2C proprietary	Proprietary	C(1)	C	C
		Android device	Standard	C(1)	CSR(5)	C
			Proprietary	C(1)	CSR(5)	C
		WP8 device	Standard	C(1)	N/A	C
			Proprietary	C(1)	N/A	N/A(3)

Legend:

C: Compatible

CSR: Compatible with strong restrictions

N/A: Not applicable

1. Proprietary data in proprietary NDEF record(s) (See *Section 4.2*).
2. Proprietary data may be corrupted by overlapping write operation.
3. Proprietary commands/files cannot be invoked/accessed in WP8 environment; access to NDEF data only.
4. Standard NDEF data writeable with proprietary application.
5. Not NFC Forum compliant.

Appendix A Example: NDEF file read sequence over RF interface

A.1 M24SR tag configuration:

- 64 K-bit EEPROM size
- NDEF file contains:
  - Record #1: URL http://www.st.com/nfc-rfid
  - Beyond NDEF message, @ address 0x1000: Proprietary data, with raw data = 0x4D 0x32 0x34 0x53 0x52 0x20 0x70 0x72 0x6F 0x70 0x72 0x69 0x65 0x74 0x61 0x72 0x79 0x20 0x64 0x61 0x74 0x61 (= String “M24SR proprietary data”)
- NDEF file is locked in Read-Only

A.2 Format of the below represented frames:

- “Listen, RF Collision Avoidance, Technology Detection, Collision Resolution” and “Device Activation” activities are supposed to be done
- ISODEP protocol bytes (PCB, CRC)
- NFC Forum Type 4 Tag commands and responses (C-APDU, R-APDU)

Table 4. NDEF file read sequence over RF interface

Select NFC T4 application	PCB	Command (C-APDU)												CRC		
		CLA	INS	P1	P2	Lc	Data							Le		
	0x02	0x00	0xA4	0x04	0x00	0x07	0xD2	0x76	0x00	0x00	0x85	0x01	0x01	0x00	0x35	0xC0
M24SR answer	PCB	Response (R-APDU)		CRC												
		Sw1	Sw2													
	0x02	0x90	0x00	0xF1	0x09											

Select CC file	PCB	Command (C-APDU)							CRC	
		CLA	INS	P1	P2	Lc	Data			
	0x03	0x00	0xA4	0x00	0x0C	0x02	0xE1	0x03	0xD2	0xAF
M24SR answer	PCB	Response (R-APDU)		CRC						
		Sw1	Sw2							
	0x03	0x90	0x00	0x2D	0x53					

Table 4. NDEF file read sequence over RF interface (continued)

Read CC file length	PCB	Command (C-APDU)					CRC																	
		CLA	INS	P1	P2	Le																		
	0x02	0x00	0xB0	0x00	0x00	0x02	0x40	0x79																
M24SR answer	PCB	Response (R-APDU)					CRC																	
		Data				Sw1	Sw2																	
	0x02	0x00	0x0F	0x90	0x00	0x44	0x45																	

Read CC file	PCB	Command (C-APDU)					CRC																	
		CLA	INS	P1	P2	Le																		
	0x03	0x00	0xB0	0x00	0x00	0x0F	0xA5	0xA2																
M24SR answer	PCB	Response (R-APDU)																				CRC		
		Data																		Sw1	Sw2			
	0x03	0x00	0x0F	0x20	0x00	0xF6	0x00	0xF6	0x04	0x06	0x00	0x01	0x20 (1)	0x00 (1)	0x00	0x80 (2)	0x90	0x00	0x45	0xFF				

Table 4. NDEF file read sequence over RF interface (continued)

Select NDEF file	PCB	Command (C-APDU)							CRC	
		CLA	INS	P1	P2	Lc	Data			
	0x02	0x00	0xA4	0x00	0x0C	0x02	0x00	0x01	0x3E	0xFD
M24SR answer	PCB	Response (R-APDU)		CRC						
		Sw1	Sw2							
	0x02	0x90	0x00	0xF1	0x09					

Read NDEF message length	PCB	Command (C-APDU)					CRC	
		CLA	INS	P1	P2	Le		
	0x03	0x00	0xB0	0x00	0x00	0x02	0x40	0x79
M24SR answer	PCB	Response (R-APDU)				CRC		
		Data		Sw1	Sw2			
	0x03	0x00	0x14	0x90	0x00	0x33	0xE2	

Table 4. NDEF file read sequence over RF interface (continued)

Read NDEF message	PCB	Command (C-APDU)					CRC														
		CLA	INS	P1	P2	Le															
	0x02	0x00	0xB0	0x00	0x02	0x14	0x6C	0x3B													
M24SR answer	PCB	Response (R-APDU)																			…
		Data																			…
	0x02	0xD1 (3)	0x01 (3)	0x10 (3)	0x55 (3)	0x01 (3)	0x73 (3)	0x74 (3)	0x2E (3)	0x63 (3)	0x6F (3)	0x6D (3)	0x2F (3)	0x6E (3)	0x66 (3)	0x63 (3)	0x2D (3)	0x72 (3)	0x66 (3)	0x69 (3)	…
	…	Response (R-APDU)			CRC																
	…	Data	Sw1	Sw2																	
	…	0x64 (3)	0x90	0x00	0x6F	0xAE															
Read beyond NDEF message	PCB	Command (C-APDU)					CRC														
		CLA	INS	P1	P2	Le															
	0x03	0xA2 (4)	0xB0 (4)	0x10 (4)	0x00 (4)	0x16	0x3C	0x56													

Table 4. NDEF file read sequence over RF interface (continued)

M24SR answer	PCB	Response (R-APDU)																		…
		Data																		…
	0x03	0x4D (4)	0x32 (4)	0x34 (4)	0x53 (4)	0x52 (4)	0x20 (4)	0x70 (4)	0x72 (4)	0x6F (4)	0x70 (4)	0x72 (4)	0x69 (4)	0x65 (4)	0x74 (4)	0x61 (4)	0x72 (4)	0x79 (4)	0x20 (4)	0x64 (4)	…
	…	Response (R-APDU)					CRC													
	…	Data			Sw1	Sw2														
	…	0x61 (4)	0x74 (4)	0x61 (4)	0x90	0x00	0x31	0xFB												

Deselect	PCB	CRC		
	0xC2	0xE0	0xB4	
M24SR answer	PCB	CRC		
	0xC2	0xE0	0xB4	

1. 64 K-bit EEPROM size
2. NDEF file is locked in Read-Only
3. NDEF File Record #1
4. String “M24SR proprietary data in the area “beyond the NDEF message”

Appendix B Example: NDEF file write sequence over RF interface

B.1 M24SR tag configuration:

- 4 K-bit EEPROM size
- NDEF file contains:
  - Record #1: URL http://www.st.com/nfc-rfid
  - Record #2: Proprietary data of type “st.com:m24sr_proprietary”, with raw data = 0x4D 0x32 0x34 0x53 0x52 0x20 0x70 0x72 0x6F 0x70 0x72 0x69 0x65 0x74 0x61 0x72 0x79 0x20 0x64 0x61 0x74 0x61 (= String “M24SR proprietary data”)

B.2 Format of the below represented frames:

- “Listen, RF Collision Avoidance, Technology Detection, Collision Resolution” and “Device Activation” activities are supposed to be done
- ISODEP protocol bytes (PCB, CRC)
- NFC Forum Type 4 Tag commands and responses (C-APDU, R-APDU)

Table 5. NDEF file write sequence over RF interface

Select NFC T4 application	PCB	Command (C-APDU)													CRC		
		CLA	INS	P1	P2	Lc	Data							Le			
	0x02	0x00	0xA4	0x04	0x00	0x07	0xD2	0x76	0x00	0x00	0x85	0x01	0x01	0x00	0x35	0xC0	
M24SR answer	PCB	Response (R-APDU)			CRC												
		Sw1	Sw2														
	0x02	0x90	0x00	0xF1	0x09												

Select CC file	PCB	Command (C-APDU)						CRC		
		CLA	INS	P1	P2	Lc	Data			
	0x03	0x00	0xA4	0x00	0x0C	0x02	0xE1	0x03	0xD2	0xAF
M24SR answer	PCB	Response (R-APDU)		CRC						
		Sw1	Sw2							
	0x03	0x90	0x00	0x2D	0x53					

Read CC file length	PCB	Command (C-APDU)					CRC		
		CLA	INS	P1	P2	Le			
	0x02	0x00	0xB0	0x00	0x00	0x02	0x6B	0x7D	
M24SR answer	PCB	Response (R-APDU)				CRC			
		Data		Sw1	Sw2				
	0x02	0x00	0x0F	0x90	0x00	0x44	0x45		

Read CC file	PCB	Command (C-APDU)					CRC													
		CLA	INS	P1	P2	Le														
	0x03	0x00	0xB0	0x00	0x00	0x0F	0xA5	0xA2												
M24SR answer	PCB	Response (R-APDU)																CRC		
		Data														Sw1	Sw2			
	0x03	0x00	0x0F	0x20	0x00	0xF6	0x00	0xF6	0x04	0x06	0x00	0x01	0x02 (1)	0x00 (1)	0x00	0x00	0x90	0x00	0x9F	0x7E

Table 5. NDEF file write sequence over RF interface (continued)

Select NDEF file	PCB	Command (C-APDU)							CRC	
		CLA	INS	P1	P2	Lc	Data			
	0x02	0x00	0xA4	0x00	0x0C	0x02	0x00	0x01	0x3E	0xFD
M24SR answer	PCB	Response (R-APDU)		CRC						
		Sw1	Sw2							
	0x02	0x90	0x00	0xF1	0x09					

Erase NDEF message length	PCB	Command (C-APDU)							CRC	
		CLA	INS	P1	P2	Lc	Data			
	0x03	0x00	0xD6	0x00	0x00	0x02	0x00	0x00	0x6B	0x37
M24SR answer	PCB	Response (R-APDU)		CRC						
		Sw1	Sw2							
	0x03	0x90	0x00	0x2D	0x53					

Write NDEF message	PCB	Command (C-APDU)																		…
		CLA	INS	P1	P2	Lc	Data													…
	0x02	0x00	0xD6	0x00	0x02	0x45	0x91 (2)	0x01 (2)	0x10 (2)	0x55 (2)	0x01 (2)	0x73 (2)	0x74 (2)	0x2E (2)	0x63 (2)	0x6F (2)	0x6D (2)	0x2F (2)	0x6E (2)	0x66 (2)	…
	…	Command (C-APDU)																		…
	…	Data																		…
	…	0x63 (2)	0x2D (2)	0x72 (2)	0x66 (2)	0x69 (2)	0x64 (2)	0x54 (3)	0x18 (3)	0x16 (3)	0x73 (3)	0x74 (3)	0x2E (3)	0x63 (3)	0x6F (3)	0x6D (3)	0x3A (3)	0x6D (3)	0x32 (3)	0x34 (3)	…
	…	Command (C-APDU)																		…
	…	Data																		…
	…	0x73 (3)	0x72 (3)	0x5F (3)	0x70 (3)	0x72 (3)	0x6F (3)	0x70 (3)	0x72 (3)	0x69 (3)	0x65 (3)	0x74 (3)	0x61 (3)	0x72 (3)	0x73 (3)	0x4D (3)	0x32 (3)	0x34 (3)	0x53 (3)	0x52 (3)	…
	…	Command (C-APDU)																	CRC	
	…	Data																		
	…	0x20 (3)	0x70 (3)	0x72 (3)	0x6F (3)	0x70 (3)	0x72 (3)	0x69 (3)	0x65 (3)	0x74 (3)	0x61 (3)	0x72 (3)	0x79 (3)	0x20 (3)	0x64 (3)	0x61 (3)	0x74 (3)	0x61 (3)	0x7C	0xF0	

Table 5. NDEF file write sequence over RF interface (continued)

M24SR answer	PCB	Response (R-APDU)		CRC							
		Sw1	Sw2								
	0x02	0x90	0x00	0xF1	0x09						

Write NDEF message length	PCB	Command (C-APDU)							CRC		
		CLA	INS	P1	P2	Lc	Data				
	0x03	0x00	0xD6	0x00	0x00	0x02	0x00	0x45	0xC2	0x22	
M24SR answer	PCB	Response (R-APDU)		CRC							
		Sw1	Sw2								
	0x03	0x90	0x00	0x2D	0x53						

Deselect	PCB	CRC									
	0xC2	0xE0	0xB4								
M24SR answer	PCB	CRC									
	0xC2	0xE0	0xB4								

1. 4 K-bit EEPROM size
2. NDEF File Record #1
3. String “M24SR proprietary data”

Appendix C Example: NDEF file read sequence over I²C protocol

C.1 M24SR tag configuration:

- 4 K-bit EEPROM size
- NDEF file contains:
  - Record #1: URL http://www.st.com/nfc-rfid

Format of the below represented frames:

- I²C specific bits sequences (Start, Stop, Ack)
- I²C protocol bytes (Device Select)
- ISODEP protocol bytes (PCB, CRC)
- NFC Forum Type 4 Tag commands and responses (C-APDU, R-APDU)

Table 6. Example: NDEF file read sequence over I²C protocol

Table 6. Example: NDEF file read sequence over I^{2}C protocol (continued)

Read CC file length		Device Select	PCB	Command (C-APDU)					CRC		
				CLA	INS	P1	P2	Le			
	S	0xAC	0x02	0x00	0xB0	0x00	0x00	0x02	0x6B	0x7D	P
M24SR answer		Device Select	PCB	Response (R-APDU)				CRC			
				Data		Sw1	Sw2				
	S	0xAD	0x02	0x00	0x0F	0x90	0x00	0x44	0x45	P	

Read CC file		Device Select	PCB	Command (C-APDU)					CRC										
				CLA	INS	P1	P2	Le											
	S	0xAC	0x03	0x00	0xB0	0x00	0x00	0x0F	0xA5	0xA2	P								
M24SR answer		Device Select	PCB	Response (R-APDU)															…
				Data															…
	S	0xAD	0x03	0x00	0x0F	0x20	0x00	0xF6	0x00	0xF6	0x04	0x06	0x00	0x01	0x02	0x00	0x00	0x00	…
			…	Response (R-APDU)		CRC													
			…	Sw1	Sw2														
			…	0x90	0x00	0x9F	0x7E	P											

Table 6. Example: NDEF file read sequence over I²C protocol (continued)

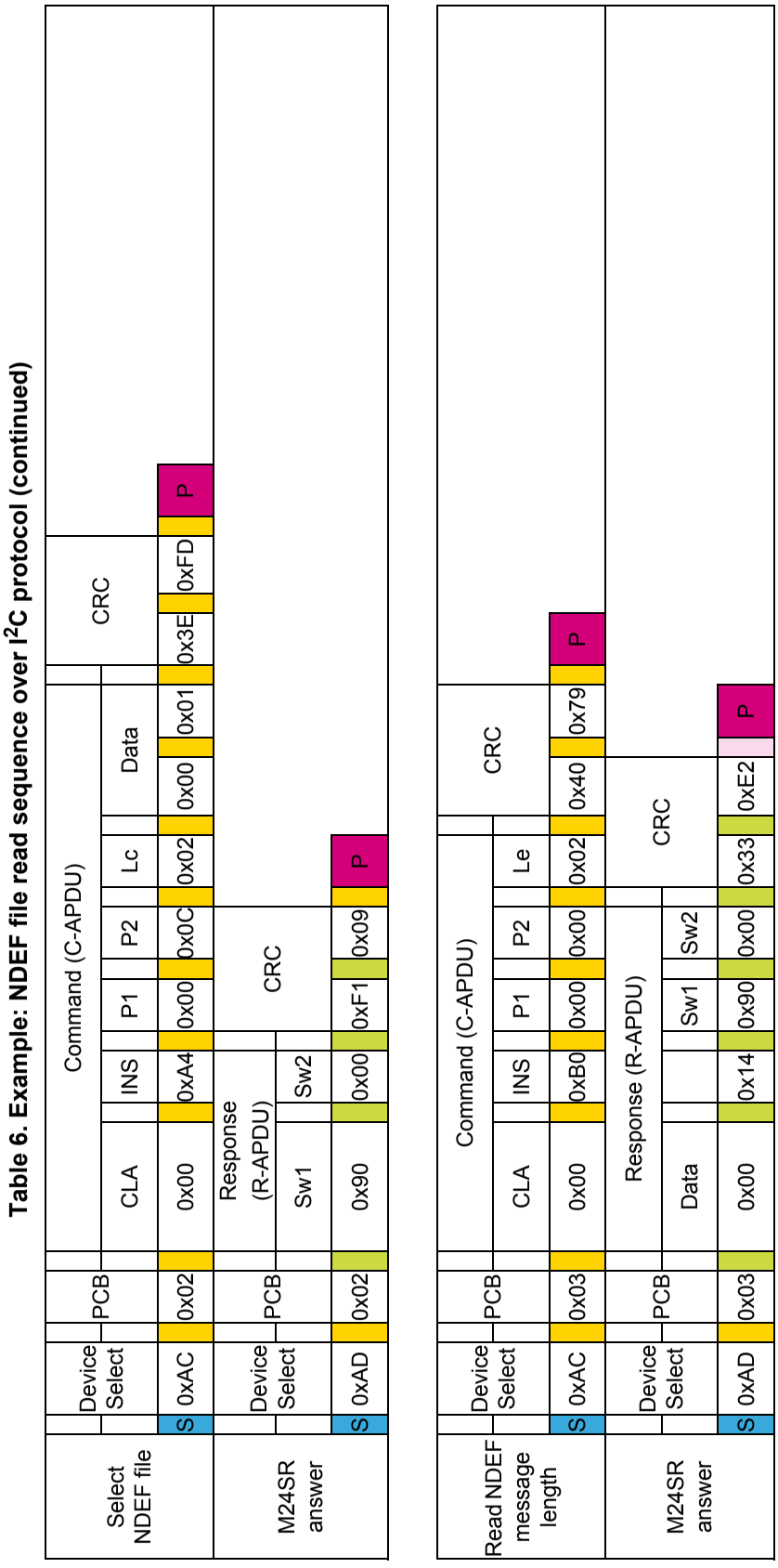
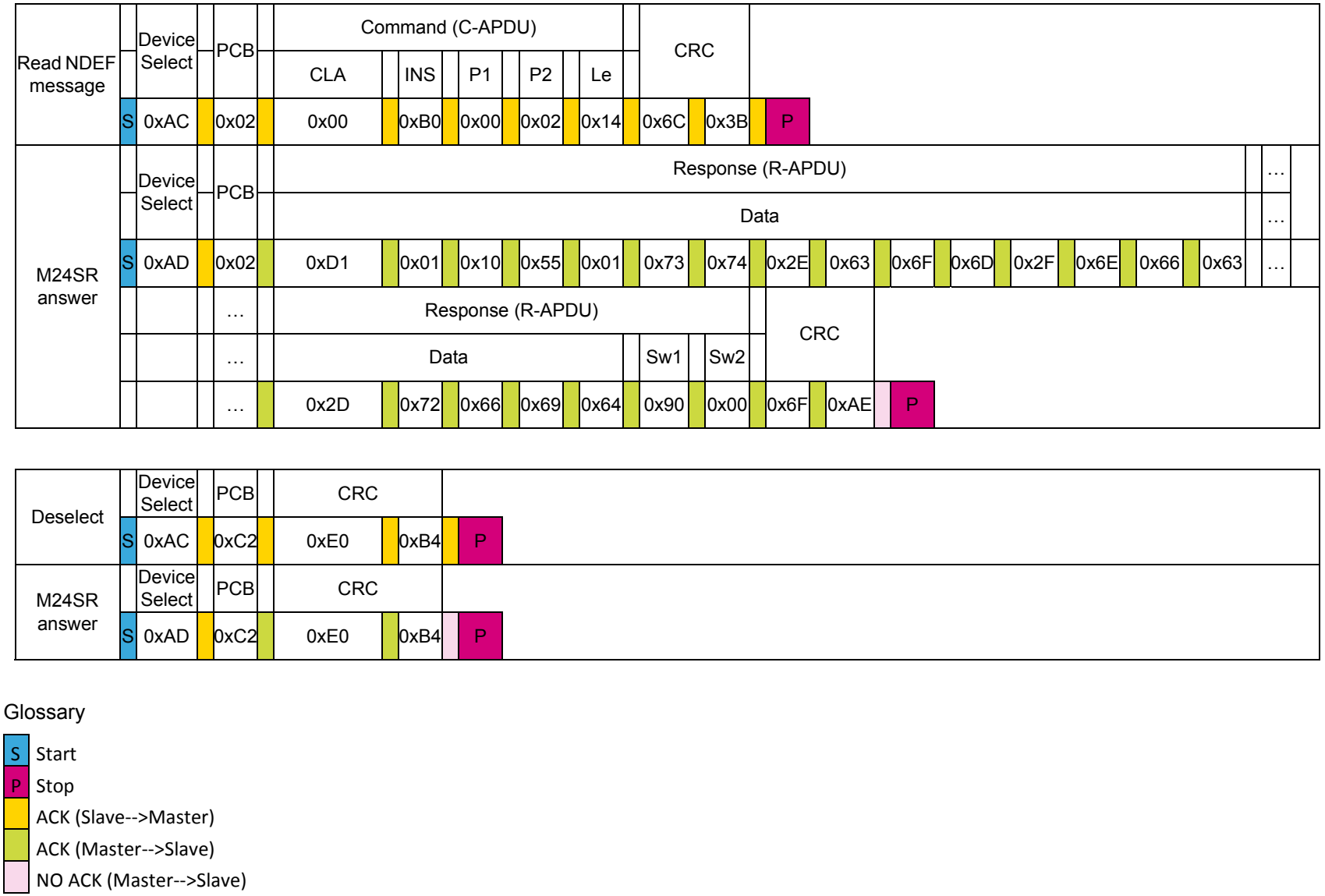


Table 6. Example: NDEF file read sequence over I²C protocol (continued)



Glossary

- S Start
- P Stop
- ACK (Slave-->Master)
- ACK (Master-->Slave)
- NO ACK (Master-->Slave)

6 Reference documents

1. NFC Forum “Digital Protocol” specification, v1.0
2. NFC Forum “Type 4 Tag Operation” specification, v2.0
3. NFC Forum “NFC Data Exchange Format (NDEF)” specification, v1.0
4. NFC Forum “NFC Record Type Definition (RTD)” specification, v1.0
5. NFC Forum “SmartPoster RTD” specification, v1.0
6. Active M24SR products datasheets.

7 Revision history

Table 7. Document revision history

Date	Revision	Changes
28-Feb-2014	1	Initial release.
27-Mar-2017	2	Updated: – *Table 4: NDEF file read sequence over RF interface* – *Table 5: NDEF file write sequence over RF interface* – *Table 6: Example: NDEF file read sequence over I2C protocol*