exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

Mitsubishi Heavy Industries (MHI) Air Condition X/Y Bus

Reverse Engineering Project Report

#2018-08-13-001

- DRAFT -

exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

Document Revisions

Version	Author	Date	Change
0.1	Markus Fenske	2018-08-17	Initial Draft

exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

Table of Contents

exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

1 Project Scope

...

- Used material:
- FDUM
- RC-E5
- Saleae
- Custom Circutry as described in Appendix A
- Software von Arduino Appendix B

...

exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

2 Results

2.1 Electrical characteristics

The signals are transmitted over a two wire bus, with terminals named X and Y, therefore called the X/Y bus. If no signals are transmitted, it supplies a voltage of 13 - 18 Volts (in our setup we measured 16.9 volts) and sources a current of at least 80 mA (in our setup we draw 5 mA max.), which attached devices may use for a parasitic power supply.[1] It includes a short circuit protection[2]

1 Electrical characteristics, as used from the a third party accessory (MH-RC-KNX-1i), see page 4. https://www.intesisbox.com/intesis/product/media/intesisbox_mh-rc-knx-1i_datasheet_en.pdf?v=2.1

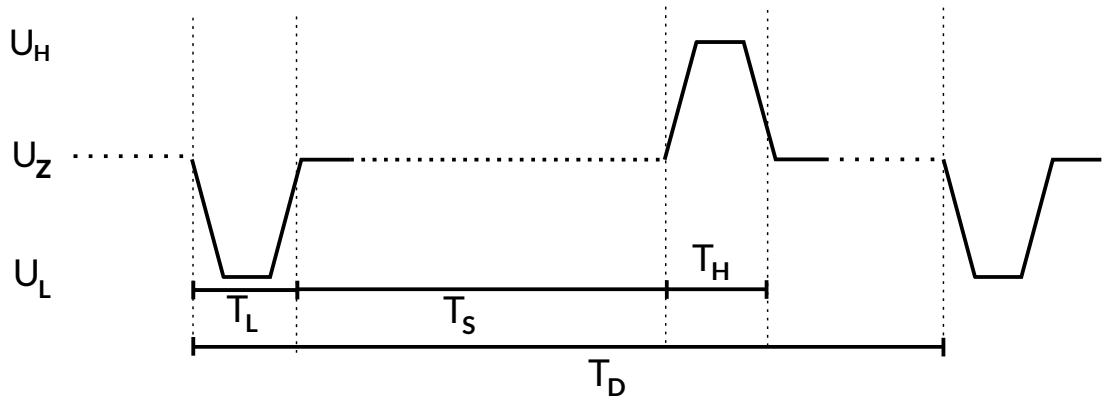
2 Error code E1 as described on page 129 (German only) http://downloads.klimacorner.de/dateien/Technisches_Handbuch_S-Serie.pdf

exablue GmbH
Empelder Str. 1a
30952 Ronnenberg
https://www.exablue.de/

2.2 Symbols

Signals on the bus are transmitted using a bipolar encoding. Each symbol is started by a low pulse and ended by a high pulse. The timing between the low and high pulse varies in 8 steps, therefore transmitting 3 bits in each symbol.

		High [V]	Typ. [V]	Low [V]
U_H	High Pulse Voltage	13.22	12	10.73
U_Z	Supply Voltage	17.02	17	16.75
U_L	Low Pulse Voltage	21.81	21	20.49



		High [µs]	Typ. [µs]	Low [µs]
T_L	Low Pulse Duration	60	52	30
T_S	Symbol Encoding Duration	$104n$ - 52 where n = {1, 2, ..., 8}		
T_H	High Pulse Duration	60	52	30
T_D	Symbol Distance Duration	1248	1248	1248

Each symbol encodes the following bits, depending on the temporal distance between the rising edge of the low pulse and the rising edge of the high pulse (T_L+T_S):

Timing	Encoded bit pattern

104 µs	000
208 µs	001
312 µs	010
416 µs	011
520 µs	100
624 µs	101
728 µs	110
832 µs	111

2.3 Bytes

3 Symbols encode a byte. The first and the second symbol encode 3 bits each, whole the last symbol encode the last 2 bits. The last symbol includes the MSB, while the first symbol includes the LSB.

For example, the value 0x23 (00100011) is encoded as follows:

0	Symbol 3 (last)	000 = 104 µs
0		
1	Symbol 2	100 = 520 µs
0		
0		
0	Symbol 1 (first)	011 = 416 µs
1		
1		

Therefore in order to send 0x23, the following sequence has to be followed:

- a 52 µs low pulse
- a 364 µs wait
- a 52 µs high pulse
- a 780 µs wait (concluding Symbol 1)
- a 52 µs low pulse
- a 468 µs wait
- a 52 µs high pulse

- a 676 µs wait (concluding Symbol 2)
- a 52 µs low pulse
- a 52 µs wait
- a 52 µs high pulse
- a 1092 µs wait (concluding Symbol 3)

2.4 Frames

Each party on the bus transmits data consisting of frames of 16 bytes (each consisting of 48 symbols). These take 59.956 ms to be sent. We found that the time of silence between two frames is typically 300 ms and does not fall below 250 ms. The maximum amount has been found during system power up where typically the system takes 30 seconds to start sending date.

Bytes 1:2: Receiver address

Those bytes encode the receiver address. In our setup, the following address were used on the bus:

0x0007: FDUM71VF air condition module

0x8001: RC-E5 wired air condition remote control

Bus Scanning

The master on the bus is the RC-E5, which continuously tries to enumerate other devices on the bus by scanning the address range. It continues with 0x0107, increasing the value by 0x1000 four times until 0x3107, then adding 0x0100 and starting the inner loop again. This goes on until 0x3f07 is reached. Then it continues with 0x0017..0x0f17, 0x0027..0x0f27, ..., 0x0f07..0x0ff7. Then 0x1ff7, 0x2ff7, 0x3ff7 are scanned, but with a different data block attached. Then the cycle starts again. It scans one device roughly every 10 seconds (scanning is faster after power loss), so the cycle takes about 29 minutes.

Byte 3: Mode of operation

This byte controls the mode of operation. Available modes are cooling (unit is used for cooling only), heating (unit is used for heating only), fan (compressor is switched off, fan only), auto (cooling and heating combined to constantly hold the selected temperature and dehumidify (compressor is cycled to dehumidify the air. collected water is pumped out using the drain pump). One bit is used to turn the unit on and off.

Bit 1: On/off	0: Unit turned off 1: Unit turned on
Bits 2-7: Mode	010101: Cooling only

	010111: Fan only 111001: Heating only 010001: Auto 010011: Dehumidify
Bit 8: Unknown	1: Unknown (default) 0: Unknown (not seen)

Byte 4: Fan speed

We found the following values.

0x98	Level 1
0x99	Level 2
0x9a	Level 3
0xaa	Off

Byte 5: Target temperature

This value reflects the value set in the remote control. The RC sets the temperature, the AC echos the value back.

The conversion formula is as follows:

(input-0x80)/2.0 = Temperature in °C

Byte 6: Unknown temperature

This value is send by the AC. In a frame send by the RC, this value is set to 0xff.

It seems to reflect a temperature. However, using the conversion formula we found implausible values (5.5 °C in a summer night with an outside temperature not below 15 !°C without the unit running). We are unsure if this temperature is measured at the inside or outside unit and how it converts.

Byte 7-15: Datablock

FIXME: Here be dragons.

Byte 16: Checksum

This value is calculated by summing the previous bytes.

checksum = sum(byte1 ... byte15) % 0xff