**Software Requirements Specification (SRS)**

For projects

**1531 (KU FE Module)**

Version: 1.0

|  |
| --- |
| **RESTRICTION OF USE, DUPLICATION, OR**  **DISCLOSURE OF PROPRIETARY INFORMATION**  This document contains proprietary information that is the sole property of DSIT. The document is submitted to the recipient for his use only. By receiving this document, the recipient undertakes not to duplicate the document or to disclose in part, or in whole any of the information contained herein to any third party without receiving beforehand written permission from DSIT. |

**Table of Contents**

[1 Changes 3](#_Toc503274528)

[2 Project 1532: 4](#_Toc503274529)

[2.1 Introduction 4](#_Toc503274530)

[2.2 System block diagram 5](#_Toc503274531)

[2.3 Objective 6](#_Toc503274532)

[2.4 Synthesizer registers values 6](#_Toc503274533)

[3 EUSART 7](#_Toc503274534)

[3.1 Electrical Specification 7](#_Toc503274535)

[3.2 Host Communication Protocol 8](#_Toc503274536)

[3.2.1 Packet Structure 9](#_Toc503274537)

[3.3 Messages 9](#_Toc503274538)

[3.3.1 Messages request groups – from host to on board MCU 10](#_Toc503274539)

[3.3.2 Control Message 11](#_Toc503274540)

[3.3.3 Status Message 12](#_Toc503274541)

[3.3.4 ADC configuration Message 13](#_Toc503274542)

[3.3.5 ADC samples values message 14](#_Toc503274543)

[3.3.6 Synthesizer (Up / Down) configuration message 15](#_Toc503274544)

[3.3.1 CPLD control message 17](#_Toc503274545)

[3.3.1 DAC control 18](#_Toc503274546)

[4 Software Update Session 19](#_Toc503274547)

[4.1 Software Update Messages 20](#_Toc503274548)

[5 Hardware 21](#_Toc503274549)

[5.1 Schematic 21](#_Toc503274550)

[5.1.1 MCU unit 21](#_Toc503274551)

[5.1.2 CPLD unit 21](#_Toc503274552)

[5.2 Microcontroller 23](#_Toc503274553)

[5.3 GPIO and Analog Pin Assignments 23](#_Toc503274554)

[6 Appendix A – How to set the registers values of sensitizer ADF-5355 25](#_Toc503274555)

# Changes

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Change | Version | Date |
| Roee Zinoue | First Edition. | 1.0 | 07/01/18 |

# 

# Project 1532:

## Introduction

This document describes the SW operation of the KU FE module.

This module will output range of frequencies according to digital data that given on a serial communication channel (based on UART-422).

The on board MCU will also configure an on board synthesizer (ADF-5355) unit to output RF signals at range of:

1. From 950 – 2620 MHz to 10.95 – 11.7 GHz. (Tx RF signal).
2. From 13.75 – 14.5 GHz to 950 – 2620 MHz. (Rx RF signal).

The MCU will also control the on board synthesizer by calibrated clock. The calibration will be made via on board CPLD unit and external DAC (PLL circuit).

The module will also be able to sample 6 analog inputs: Temperature, reverse / forward / real voltage and current and stored the samples info on internal FLASH memory section at the MCU (PIC18F45K22) unit.

At a user request those ADC samples (that stored on the internal MCU flash memory) can be read via serial communication at external PC application (among other parameters like: SW version, SW state, synthesizer registers values, RF output signal, etc...).

## System block diagram



## Objective

Main object of the on board MCU (PIC18F45K22):

1. To configure the on board synthesizer (ADF-5355) to exit RF signals at the range of:
   1. Tx RF signal: From 950 – 2620 MHz to 10.95 – 11.7 GHz.
   2. Rx RF signal: From 13.75 – 14.5 GHz to 950 – 2620 MHz.
2. To sample 6 analog inputs signal like: Temperature, reverse / forward / real voltage and current.
3. To store ADC samples at internal flash memory section. And flush them at user selection (this request will be given on the serial channel).
4. To communication via serial communication (RS-422) with external host station (PC) and send / receive data.
5. To indicate the system user on system state / failures via external LEDs.

Main object of the on board CPLD unit:

1. To calibrate digital clock that is an input to the on board synthesizer unit (ADF-5355) via VCO circuit using external DAC.

## Synthesizer registers values

The registers values that configure the RF output signals are calculated on the on board MCU at each system run. Please refer to document [appendix A](#_Appendix_A_–) to see how they are calculated.

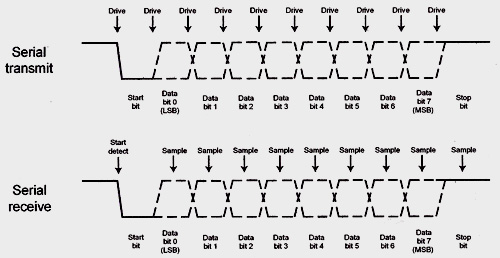
# EUSART

## Electrical Specification

The chosen communication channel with the external host is an RS422 with configuration:

1. Half Duplex – Tx / Rx channels.
2. Baud-rate of: 115200 Kb/s.
3. 8 bits, 1 start bit, 2 stop bit.
4. Hardware control, No CTS, RTS.
5. Voltage range: -7 to +7 VDC.

A detailed waveform diagram of the UART channel can be seen below:



## Host Communication Protocol

The communication protocol between host and microcontroller is a simple master/slave setup. The host always initiates communication by sending its command message. If the microcontroller receives a correct message (framing and CRC are correct), it sends back an ACK replay. If the received message is a request a data from the MCU unit, the returned data will should be treated as an ACK by the host.

If a packet is received with an error, the unit will not respond and the host should treat the sent packet as lost. The timeout for a lost packet is approx. 10ms and 10 continues lost packets.

The packets content is sent on the RS422 channel using an RFC1662 (PPP in HDLC-like Framing) inspired implementation and is protected by an 8bit CRC as required in the RFC.

Packet length is devised from the framing mechanism.

The RFC1662 specification guarantees that the framings char will not appear in the data payload by using an “escape” mechanism, each byte is in the message is compared to the framing character (0x7e) and to the escape character (0x7d) and if it is equal to one of them than the character is ‘escaped’ by inserting a 0x7d in-front of it in the message and then XORing the character with 0x20. As a byproduct of the escape method, the message length can increase by 100% + 2 framing bytes + 1 CRC byte.

The RFC1662 protocol defines the use of a CRC on the packet to ensure that a correct packet has been received. The CRC polynom used in the MFE is x^8 + x^2 + x + 1 and will be implemented via a lookup table (and a XOR calculation per byte).

The host can send up to 2000 messages per second to the MCU unit.

### Packet Structure

Description about packet structure: Each request from host to MCU will get feedback (ACK) message response.

#### From host to MCU – request packet

|  |  |  |
| --- | --- | --- |
| Byte # | Value | Details |
| 0 – 1 | **0xA5 0x5A** | Frame Start Chars. |
| 2 | **MSG\_GROUP** | Message group. |
| 3 | **MSG\_REQ** | Message request type. |
| N >= 0 |  | Number of data bytes. |
| D = N |  | Data. |
| 4 – 5 | CRC | CRC-16 code. |

Each request consist 6 fixed bytes + N bytes of data in single request frame.

#### From MCU to host – response packet

|  |  |  |
| --- | --- | --- |
| Byte # | Value | Details |
| 0 – 1 | **0xA5 0x5A** | Frame Start Chars. |
| 2 | **MSG\_GROUP** | Message group. |
| 3 | **MSG\_RESP** | Message response type. |
| N >= 0 |  | Number of data bytes. |
| D = N |  | Data. |
| 4 – 5 | CRC | CRC-16 code. |

Each response consist 6 fixed bytes + N bytes of data in single request frame.

## Messages

This section defines all messages groups transmitted from the external host to the MCU.

All messages are transmitted as hexadecimal data bytes.

Fields that are more than 1 byte long, will be passed in a little-endian format, meaning that the first byte is the least significant and last byte is most significant.

All fields in the messages below that have no explicit size defined, are 8 bit entities.

### Messages request groups – from host to on board MCU

|  |  |  |
| --- | --- | --- |
| Group ID | Group name | Short group information |
| 0x01 | Control | Host set the MCU operation parameters |
| 0x02 | Status and version | Host get the MCU operation status and run-time statistics. |
| 0x03 | ADC configuration | Host set the MCU ADC configuration parameters. |
| 0x04 | ADC samples values | Host get the MCU ADC sampled values. |
| 0x05 | Synthesizer (Up / Down) configuration | Host get or set the Synthesizer up registers configuration parameters. |
| 0x06 | CPLD control | Host get or set the CPLD operation settings and run-time statistics. |
| 0x07 | DAC control | Host set the external CPLD DAC configuration. |

### Control Message

Group ID: 0x01

The control message group allows the external host to control the operation of the MCU.

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | MCU reset | Reset the MCU. |
| 0x2 | CPLD reset | Reset the CPLD. |
| 0x3 | P.A Enable | Enable the P.A. |
| 0x4 | RS-422 baud selection | Set the MCU baud-rate speed. |
| 0x5 | Test LEDs | Insert the 2 connected Leds to blink mode test (the Leds will blink for 10 sec at 1 sec toggle time interval). |

Baud-rate selection data:

|  |  |
| --- | --- |
| Data Byte value | Baud-rate [bits/sec] |
| 0x1 | 9600 |
| 0x2 | 19200 |
| 0x3 | 57600 |
| 0x4 | 115200 |
| 0x5 | 921600 |

**Response packet info (MSG\_RESP):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size | Description |
| 0x3 | P.A Enable OK | 0 | ACK packet response bit to indicate that the MCU set this option |
| 0x4 | Set serial baud-rate OK |  | ACK packet response bit to indicate that the MCU set this option |
| 0x5 | Test LEDs ok | 0 | Leds test start indication. |

### Status Message

Group ID: 0x02

The status message group allows the external host to read statistic from the on board mcu such as run time and the on board MCU FW version.

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | Get run time (mins). | Request from the on board MCU the run time (power on) in time format (HH:MM:SS). |
| 0x2 | Get MCU FW programed date. | Request from the on board MCU the date the last programming was preformed (YY:MM:DD) |
| 0x3 | Set the MCU FW programed date. | Set the on board MCU the F.W date in format (YY:MM:DD) |
| 0x4 | Get MCU serial number. | Request from the on board MCU the unique serial number (6 Bytes) that was written in the EEPROM memory during calibration progress. |
| 0x5 | Set the MCU serial number. | Set the on board MCU the unique serial number (6 Bytes), this number will set on the MCU EEPROM memory section. |

**Response packet info (MSG\_RESP):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size (Bytes) | Description |
| 0x1 | MCU run time value. | 6 | MCU run time in format: HH:MM:SS. |
| 0x2 | MCU programed date value. | 6 | The last programed date in format: YY:MM:DD. |
| 0x3 | Set MCU programed date OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x4 | MCU card serial number value. | 6 | MCU serial. |
| 0x5 | Set MCU serial number OK | 0 | ACK packet response bit to indicate that the MCU set this option. |

### ADC configuration Message

Group ID: 0x03

The ADC configuration Message group allows the external host to set the internal on board MCU ADC unit registers configuration values.

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | ADC enable | Enable / disable the internal MCU ADC unit. |
| 0x2 | ADC Vref negative | Select the ADC negative voltage reference source. |
| 0x3 | Analog Channel sample Select | Set ADC to sample single channel (CH: 0 -27) or range of channels (set by host application) in circular mode. |
| 0x4 | ADC Vref positive | Select the ADC positive voltage reference source. |
| 0x5 | Conversion Result Format | Select if the ADC conversion results are Right justified or Left justified. |
| 0x6 | Conversion Clock Select | Select the ADC clock source. |

ADC channel selection:

|  |  |
| --- | --- |
| Data Byte value | Channel name |
| 0x1 | Pre-Amp Temperature. |
| 0x2 | Reverse analog voltage. |
| 0x3 | Reverse analog voltage. |
| 0x4 | Forward analog voltage. |
| 0x5 | Input to system analog voltage. |
| 0x6 | Current analog voltage. |

**Response packet info (MSG\_RESP):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size (Bytes) | Description |
| 0x1 | Set ADC enable OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x2 | Set ADC Vref negative value OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x3 | Set Vref positive value OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x4 | Set Conversion Result Format OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x5 | Set Conversion Clock Select | 0 | ACK packet response bit to indicate that the MCU set this option. |

### ADC samples values message

Group ID: 0x04

The ADC samples values message group allows the external host to get N samples from the internal on board MCU ADC unit.

This N values read from the internal flash memory.

Please note that the sample precision result (Sample resolution) define as:

At each conversion round the host define first the number of sampled results to receive (Samples >= 0) from the internal MCU ADC unit.

If there is more than 1 samples request, the samples will separate with chars @@. Each sample results size is define as 10 bits (3 Bytes).

At the end of the conversion round the MCU will transmit frame that consist all the raw data that he sampled (this raw data stored at flash memory section during the conversion round till finish. Then the MCU will erase this memory section and prepare it to store new data).

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | Request N bytes of samples (N >= 0) | Host request N samples from the internal MCU ADC unit. |
| 0x2 | Multiple factor | Host request N samples from the internal MCU ADC unit, all samples will multiple by M factor (M >= 0 && M <= 0xFF). |
| 0x3 | Divide factor | Host request N samples from the internal MCU ADC unit, all samples will device by D factor (M >= 0 && M <= 0xFF). |

**Response packet info (MSG\_RESP):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size (Bytes) | Data Range (Hexadecimal) | Description |
| 0x1 | N samples | B >= 0 | ( \* M or ( | The return samples raw data |

### Synthesizer (Up / Down) configuration message

Group ID: 0x05

The Synthesizer (up / down) configuration message group allows the external host to set the configuration registers values of synthesizer (ADF-5355) up / down (Rx / Tx) units.

To set the output frequency from the synthesizer unit the INT and FRAQ need first to set.

As noted in [Appendix B](#_Appendix_B_–) of this document.

**Request packet info (MSG\_REQ):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description | |
| synthesizer down | | | |
| 0x01 | Set synthesizer down AUTOCAL value | | Host set the AUTOCAL register value of the up synthesizer down. |
| 0x02 | Set synthesizer down prescaler(PR1) value | | Host set the prescaler register value of the up synthesizer down. |
| 0x03 | Set synthesizer down INT value | | Host set the INT register value of the up synthesizer down. |
| 0x04 | Set synthesizer down FRAC1 value | | Host set the FRAC1 register value of the synthesizer down. |
| 0x05 | Set synthesizer down FRAC2 value | | Host set the FRAC2 register value of the synthesizer down. |
| 0x06 | Set synthesizer down MOD1 value | | Host set the MOD1 register value of the synthesizer down. |
| 0x07 | Set synthesizer down MOD2 value | | Host set the MOD2 register value of the synthesizer down. |
| 0x07 – 0x0F | Reserved bytes | | Reserved |
| Synthesizer up | | | |
| 0x10 | Set synthesizer up AUTOCAL value | | Host set the AUTOCAL register value of the up synthesizer up. |
| 0x11 | Set synthesizer up prescaler value | | Host set the prescaler register value of the up synthesizer up. |
| 0x12 | Set synthesizer up INT value | | Host set the INT register value of the up synthesizer up. |
| 0x13 | Set synthesizer up FRAC1 value | | Host set the FRAC1 register value of the synthesizer up. |
| 0x14 | Set synthesizer up FRAC2 value | | Host set the FRAC2 register value of the synthesizer up. |
| 0x15 | Set synthesizer up MOD1 value | | Host set the MOD1 register value of the synthesizer up. |
| 0x16 | Set synthesizer up MOD2 value | | Host set the MOD2 register value of the synthesizer up. |
| 0x07 – 0x0F | Reserved bytes | | Reserved |

Synthesizer registers data value:

|  |  |  |  |
| --- | --- | --- | --- |
| Register name | Data bits size | Allowed range | Description |
| AC1 | **1** | **0 - 1** | 1: Disable automatic calibration  0: Enable automatic calibration |
| PR1 | **1** | **0 – 1** | 0: Prescaler = 4/5  1: Prescaler = 8/9 |
| INT | **16** |  | Refer to [Appendix B](#_Appendix_B_–) |
| FREQ1 | **24** | **0 – 0xFFFFFF** | Refer to [Appendix B](#_Appendix_B_–) |
| FREQ2 | **14** | **0 – 0x3FFF** | Refer to [Appendix B](#_Appendix_B_–) |
| MOD1 | 14 | 0 – 0x3FFF | Refer to [Appendix B](#_Appendix_B_–) |

**Response packet info (MSG\_RESP):**

Response bit indicate all the request bits at their location. If they

Set: the MCU set the option on ok.

Preset: the MCU fail to set this option.

### CPLD control message

Group ID: 0x06

The CPLD control message group allows the external host to read and set the CPLD operation settings

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | Get run time (mins). | Request from the on board MCU the CPLD run time (power on) in time format (HH:MM:SS). |
| 0x2 | Get CPLD FW programed date. | Request from the on board MCU the date the last programming on the CPLD was preformed (YY:MM:DD) |
| 0x3 | Set the CPLD FW programed date. | Set the on board CPLD the F.W date in format (YY:MM:DD) |
| 0x4 | Get CPLD serial number. | Request from the on board MCU the CPLD unique serial number (6 Bytes) that was written in the EEPROM memory during calibration progress. |
| 0x5 | Set the CPLD serial number. | Set the CPLD unique serial number (6 Bytes), this number will set on the MCU EEPROM memory section. |

**Response packet info (MSG\_RESP):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size (Bytes) | Description |
| 0x1 | MC CPLD U run time value. | 6 | CPLD run time in format: HH:MM:SS. |
| 0x2 | CPLD programed date value. | 6 | The last programed date in format: YY:MM:DD. |
| 0x3 | Set CPLD programed date OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x4 | CPLD card serial number value. | 6 | MCU serial. |
| 0x5 | Set CPLD serial number OK | 0 | ACK packet response bit to indicate that the MCU set this option. |

### DAC control

Group ID: 0x07

The DAC control message group allows the external host to set the external DAC registers values throw written word that output from the CPLD unit.

Beside the digital word to be converted to analog voltage, we can program the DAC to operate at 4 power modes. This done by settings 2 bits in the convert register:

**Request packet info (MSG\_REQ):**

|  |  |  |
| --- | --- | --- |
| Byte  MSG\_REQ value | Name | Description |
| 0x1 | Set ADC in Normal Operation mode | Host set the ADC to operate in normal voltage mode. |
| 0x2 | Set ADC in Power-Down A operation mode | Host set the ADC to operate in Power-Down (1 kΩ Load to GND) voltage mode. |
| 0x3 | Set ADC in Power-Down B operation mode | Host set the ADC to operate in Power-Down (100 kΩ Load to GND) voltage mode. |
| 0x4 | Set ADC in Power-Down C operation mode | Host set the ADC to operate in Power-Down (High Impedance Output) voltage mode. |

**Response packet info (MSG\_RESP):**

|  |  |  |  |
| --- | --- | --- | --- |
| Byte  MSG\_RESP value | Name | DATA size (Bytes) | Description |
| 0x1 | Set ADC in Normal Operation mode OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x2 | Set ADC in Normal Operation mode OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x3 | Set ADC in Normal Operation mode OK | 0 | ACK packet response bit to indicate that the MCU set this option. |
| 0x4 | Set ADC in Normal Operation mode OK | 0 | ACK packet response bit to indicate that the MCU set this option. |

# Software Update Session

The MFE software update component is a piece of software that is responsible for checking the integrity of the operational software, loading it and running it.

The software update component is also responsible for getting a new software image from the host, verify its integrity and saving it to internal flash.

To start the software update process, a power up should be performed with the PROG\_EN discrete line set to ‘0’. When PROG\_EN equals ‘0’ the software update process does not load the operational software from flash but instead waits for configuration data from the host.

The software update process will sample the PROG\_EN discrete line for approx. TBD milliseconds. If while sampling the PROG\_EN discrete its value does not equal ‘0’ the software update process will load the operational software (if it exists) and jump to it.

As in the operational application, the communication channel is an RS422 Half Duplex, 1Mb/s communication channel as described in section 5.1.

Also, the software update component also uses the RFFC1662 framing as described in section 5.3 and the message / response packet structure is the same.

To verify that the software update component has indeed loaded correctly, the host should request the version string from the MFE. The Version string form the MFE software update component is in the form of B.x where B denotes that this is a software update component version.

When starting a new flash programming operation, the software update process will start by erasing the flash sections affected by the data received, this step is destructive and data saved on that sector will be lost.

The software update component cannot update itself, in case that the software update component needs to be updated a physical access to the MFE will be needed (a PICit3 or similar debugger will be needed to program the MFE controller via ICSP).

Calibration parameters are not programmed via the software update process, and will not be effected by a software upgrade.

## Software Update Messages

As stated above, the software update component uses the same communication channel, framing and message format as the operational software, this allows the host to use the same codebase for communicating with the MFE software update component .

The MFE software update component can receive the following messages:

* Version Request.
* Set Data Line.
* Get Data Line.
* Finished Update Process.

The MFE will respond with the following responses:

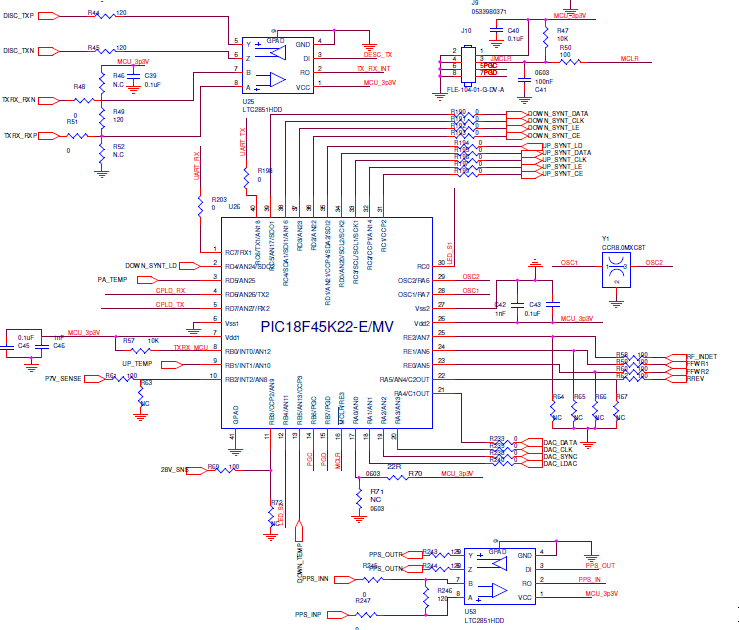
* Version Response.
* Data Line Status.
* Data Line Data.
* Ack.

### 

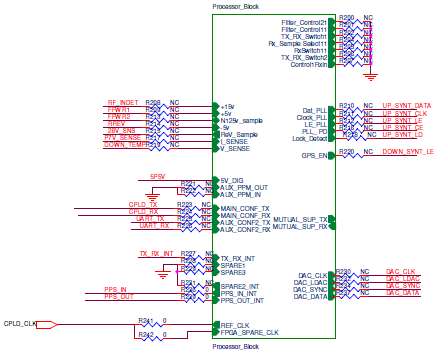
# Hardware

## Schematic

### MCU unit



### CPLD unit



## Microcontroller

|  |  |
| --- | --- |
| **Recommended PIC** | **PIC18F45K22 – 8 bit core** |
| **Operating voltage** | **3.3V** |
| **Inputs (TTL / converter)** | **TBD** |
| **Outputs (TTL / converter)** | **TBD** |
| **POR** | **Available** |
| **Internal clock** | **8MHz and up to 64Mhz** |
| **Pin count** | **40** |

* Flash Memory
  + The PIC microcontroller has 32Kbytes of internal flash memory. This memory will be used for both the software update component software and for the operational software.
* RAM Memory
  + The PIC microcontroller has 1536 bytes of internal RAM, the software will use this memory for its stack & heap.
* Peripherals support:

o Connectivity: 2-UART, 2-SPI, 2-I2C2-MSSP(SPI/I2C).

o ADC: 28 ch, 10-bit.

## GPIO and Analog Pin Assignments

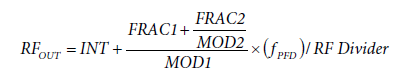
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name in document** | **PIN Name** | | **Net Name** | | **Type** |
| **MCU power, clocks and programing pins:** | | | | | |
| **3.3 VDC Enable** | | **VDD1** | **VDD1** | **Input Analog** | |
| **MCU\_3p3V** | | **RA0/AN0** | **MCU\_3p3V** | **Input Analog** | |
| **Digital ground** | | **VSS1** | **VSS1** | **Input Analog** | |
| **Program data Enable** | | **RB7/PGD** | **PGD** | **Bi-directional Discrete** | |
| **Program clock Enable** | | **RB6/PGC** | **PGC** | **Input Discrete** | |
| **System reset** | | **MCLR/RE3** | **MCLRM** | **Input Discrete** | |
| **OSC1** | | **OSC1/RA7** | **OSC1** | **Input Analog** | |
| **OSC2** | | **OSC2/RA6** | **OSC2** | **Input Analog** | |
| **RS-422 serial communication** | | | | | |
| **UART 422 RX** | **RC7/RX1** | | **UART\_RX** | **Input Discrete** | |
| **UART 422 TX** | **RC6/TX1/AN18** | | **UART\_TX** | **Output Discrete** | |
| **TXRX\_MCU** | **RB0/INT0/AN12** | | **TXRX\_MCU** | **Input Discrete** | |
| **ADC analog inputs** | | | | | |
| **RF\_INDET** | **RE2/AN7** | | **RF\_INDET** | **Input Analog** | |
| **FFWR1** | **RE1/AN6** | | **FFWR1** | **Input Analog** | |
| **FFWR2** | **RE0/AN5** | | **FFWR2** | **Input Analog** | |
| **RREV** | **RA5/AN4/C2OUT** | | **RREV** | **Input Analog** | |
| **PA\_TEMP** | **RD5/AN25** | | **PA\_TEMP** | **Input Analog** | |
| **UP\_TEMP** | **RB1/INT1/AN10** | | **UP\_TEMP** | **Input Analog** | |
| **DOWN\_TEMP** | **RB5/AN13/CCP3** | | **DOWN\_TEMP** | **Input Analog** | |
| **System status indication** | | | | | |
| **LED\_S1** | **RC0** | | **LED\_S1** | **Output Discrete** | |
| **LED\_S2** | **RB4/AN11** | | **LED\_S2** | **Output Discrete** | |
| **Synthesizer RX down** | | | | | |
| **DOWN\_SYNT\_LD** | **RD4/AN24/SDO2** | | **DOWN\_SYNT\_LD** | **Input Discrete** | |
| **DOWN\_SYNT\_DATA** | **C5/AN17/SDO1** | | **DOWN\_SYNT\_DATA** | **Output Discrete** | |
| **DOWN\_SYNT\_CLK** | **RC4/SDA1/SDI1/AN16** | | **DOWN\_SYNT\_CLK** | **Output Discrete** | |
| **DOWN\_SYNT\_LE** | **RD3/AN23** | | **DOWN\_SYNT\_LE** | **Output Discrete** | |
| **DOWN\_SYNT\_CE** | **RD2/AN22** | | **DOWN\_SYNT\_CE** | **Output Discrete** | |
| **Synthesizer TX up** | | | | | |
| **UP\_SYNT\_LD** | **RD1/AN21/CCP4/SDA2/SDI2** | | **UP\_SYNT\_LD** | **Input Discrete** | |
| **UP\_SYNT\_DATA** | **RD0/AN20/SCL2/SCK2** | | **UP\_SYNT\_DATA** | **Output Discrete** | |
| **UP\_SYNT\_CLK** | **RC3/SCL/SCL1/SCK1** | | **UP\_SYNT\_CLK** | **Output Discrete** | |
| **UP\_SYNT\_LE** | **RC2/CCP1/AN1** | | **UP\_SYNT\_LE** | **Output Discrete** | |
| **UP\_SYNT\_CE** | **RC1/CCP2** | | **UP\_SYNT\_CE** | **Output Discrete** | |
| **DAC inputs – Need change connections to CPLD** | | | | | |
| **DAC\_DATA** | **RA4/C1OUT** | | **DAC\_DATA** | **Input Discrete** | |
| **DAC\_CLK** | **RA3/AN3** | | **DAC\_CLK** | **Input Discrete** | |
| **DAC\_SYNC** | **RA2/AN2** | | **DAC\_SYNC** | **Input Discrete** | |
| **AC\_LDAC** | **RA1/AN1** | | **AC\_LDAC** | **Input Discrete** | |
| **Sensors** | | | | | |
| **28V\_SNS** | **RB3/CCP2/AN9** | | **28V\_SNS** | **Input Discrete** | |
| **P7V\_SENSE** | **P7V\_SENSE** | | **P7V\_SENSE** | **Input Discrete** | |
| **CPLD connection** | | | | | |
| **CPLD\_RX** | **RD6/AN26/TX2** | | **CPLD\_RX** | **Input Analog** | |
| **CPLD\_TX** | **RD7/AN27/RX2** | | **CPLD\_TX** | **Output Discrete** | |

# Appendix A – How to set the registers values of sensitizer ADF-5355

\*\* Please refer to unit datasheet for full details.

The Sensitizer ADF-5355 is software programmable unit which mean that each of the unit registers data is given and controlled by 12 registers that each one have control buffer of 32-bit. This registers configuration values will output from the on board MCU unit.

The main equation to calculate the output RF signal from this synthesizer is:



Where:

RFOUT is the RF output frequency.

INT is the integer division factor.

FRAC1 is the fractionality.

FRAC2 is the auxiliary fractionality.

MOD1 is the fixed 24-bit modulus.

MOD2 is the auxiliary modulus.

RF Divider is the output divider that divides down the VCO frequency.

And

*fPFD* = *REFIN* × ((1 + *D*)/(*R* × (1 + *T*)))

Where:

REFIN is the reference frequency input.

D is the REFIN doubler bit.

R is the REF reference division factor.

T is the reference divide by 2 bit (0 or 1).

All this registers values will set from the system host via serial communication channel.