Construction and Implementation of Tracking System for the ISAAC Sounding Rocket Experiment

Emil Lundkvist and Markus Silberstein Hont

Abstract—The ISAAC rocket experiment is part of the REXUS–15 payload. It has two free-falling units performing infrared spectroscopy between them. One of the free-falling units will actively track the other in the sky so as to enable continuous measurements. In this thesis report, the design of this tracking solution is described. The general concept of the tracking process is outlined, upon which the conceptuals as well as the hardware, software and firmware of each subsystem are explained. Tests of respective subsystems are conducted and conclusions are drawn pertaining to the accuracy and suitability of each subsystem for the tracking system.

Index Terms—Tracking system, Sounding Rocket Experiment, VHDL, REXUS, I\textsuperscript{2}C, FPGA, CMOS Camera Sensor, Sun Sensing, Hardware Design, Software Design

I. INTRODUCTION

ISAAC - Infrared Spectroscopy to Analyze the middle Atmosphere Composition - is a sounding rocket experiment developed at the Division of Space and Plasma Physics at the School of Electrical Engineering and the Division of Mechanics at the School of Engineering Sciences within KTH, together with the Department of Meteorology at Stockholm University (MISU).

ISAAC is scheduled to be launched in March 2014 onboard the REXUS–15 sounding rocket. REXUS/BEXUS is a student program realized under a bilateral agreement between the German Aerospace Center (DLR) and the Swedish National Space Board (SNSB) [1].

A. Scientific Purpose of the ISAAC Project

It is widely known that CO\textsubscript{2} in the Earth’s atmosphere has an influence on our planet’s climate, being a major contribution to the greenhouse effect. But less is known about its function at higher altitudes, where it is responsible for strong radiative cooling [2]. One issue is that not much information is available on the CO\textsubscript{2} concentration in the middle atmosphere (altitudes of 12-80 km). The middle atmosphere is particularly interesting since previous research suggests a decrease in CO\textsubscript{2} concentration starting at altitudes of around 70-75 km and better data regarding this would help improve our understanding of the climate of the Earth [3].

B. Experiment Overview

Two FFUs, referred to as ISAAC–Tx and ISAAC–Rx, are located inside the RMU, which in turn is placed on the REXUS rocket. The FFUs are ejected from the RMU when the rocket reaches above the dense atmosphere, at an altitude of approximately 60 km [4]. After ejection, both FFUs are spinning around their own cylindrical central axes. Throughout its free fall, the ISAAC–Tx is continuously and actively tracked by the ISAAC–Rx which provides the capability of performing IR spectroscopy between them: the change in intensity between emission from the ISAAC–Tx and reception at the ISAAC–Rx is measured and thereby the CO\textsubscript{2} concentration can be calculated in post-processing. (Refer to Fig. 1.)
C. Scope

The goal of this thesis report is to describe the design, the construction and the implementation of the tracking system which is crucial for the IR spectroscopy to take place. In Section II, a general overview of the tracking process is outlined and its different sub-systems are described, both conceptually and theoretically, and how they interact with each other in order to achieve tracking. Furthermore, the tracking algorithm is described on a theoretical level. Section III features the in-depth description of the hardware functionality, how the specific components are chosen and how they function electronically. Section IV features the actual implementation of the tracking system. Circuit designs and layouts of the PCBs used for the tracking are described. The in-flight data processing and control algorithms which are executed by the ISAAC–Rx in order to locate the ISAAC–Tx are presented in detail in terms of the programming. Section V describes test procedures carried out, including reasons for testing as well as the ways in which they were performed. Section VI presents the results from these tests and from the working process of the thesis project. Section VII contains discussion, both concerning results from tests as well as the thesis project at whole.

II. TRACKING SYSTEM OVERVIEW

In order to perform spectroscopy, the aperture on the ISAAC–Rx must be facing the IR light source on the ISAAC–Tx. This is the sole purpose of the tracking system and the concept of its execution is described below.

A. Tracking System Concept

The ISAAC–Tx serves as a ‘passive’ unit in the sense that it will be completely unaware of the whereabouts of the ISAAC–Rx, which in turn is the unit entirely responsible for carrying out the tracking. When the FFUs are ejected from the RMU, they are spinning at the rate of the rocket [1]. In order to enable the aperture on the ISAAC–Rx to be fixedly pointed at the ISAAC–Tx, the Rx unit is divided into two sub-units: the RxSU, which contains the aperture and all the tracking hardware, and the CU, which contains a GPS system for post-flight recovery, amongst other things which are not relevant for the tracking. In between the sub-units is mounted a motor which allows the RxSU to assume a non-spinning state in the common reference frame of the FFUs, which fixes the aperture horizontally. Vertical adjustments are made by tilting a mirror which directs the IR light onto a sensor. Together, the repeated vertical and horizontal adjustments allow the spectroscopy to be carried out continuously throughout the data gathering stage in Fig 1.

The amounts of these adjustments are calculated using data from three different kinds of sensors mounted onto the RxSU:

1) Sun Sensors: The initial stage after the ejection of the FFUs is for the ISAAC–Rx to roughly approximate the location of the Tx unit in the sky. This is done by relating the axis of rotation to the position of the sun using a sun sensing system (the electronics of which is described in further detail in section III).

2) CMOS Camera: Mounted (evenly spaced) onto the skin of the ISAAC–Tx are 20 LEDs (placed in pairs). The purpose of these is to increase the visibility of the Tx unit in the sky. Once the approximate location of the ISAAC–Tx has been calculated using the sun sensing system, the camera more accurately determines where to aim the aperture. Images of the LEDs are captured and analyzed (refer to section IV) after which the rotation speed and mirror tilt angle can be appropriately adjusted.

3) Angular Rate Sensor: In order to assume the non-spinning state, the control algorithm needs data input concerning the total spinning rate of the RxSU, and it is from the angular rate gyro that these data come. Furthermore, the image processing algorithm (refer to section II-C) uses data from this sensor to predict movement of the LEDs across the field of view.

Figure 2 describes the overall concept of the tracking system.

![Fig. 2: Conceptual diagram of the tracking system](image)

B. Sub-System Theory

1) Sun Sensors: The FFUs are ejected from the REXUS rocket at a given time, and therefore they follow a certain known trajectory relative to the sun. This fact matters to ISAAC for two reasons. Firstly, the position of the sun is important when, after ejection, the ISAAC-Rx attempts to make a first rough approximation as to the whereabouts of the ISAAC-Tx using its onboard sun sensors. Secondly, the performance of the tracking camera would be severely degraded, should it be ‘blinded’ by the sun. This also needs to be accounted for. There are four sun sensors mounted evenly spaced onto the skin of the ISAAC-Rx. They measure the intensity of incident sunlight at any given sampling time. When ejection occurs, the FFUs are spinning at a rate of approximately 4 Hz [1], which allows for a periodic behavior in the intensity readouts from each sun sensor. If all the sensors are sampled in phase with each other, one can compare differences in intensity between them and thus interpolate the relative position of the sun. These data are then used to control the rate of the de-spinning motor mounted between the RxSU and its CU in such a way that the optical system is horizontally aligned with the ISAAC-Tx.
The goal of the sun sensing system is to control the spinning in such a way that one sun sensor (referred to as SS1 in Fig. 3) is constantly aligned with the sun in the plane of the RxSU. This is achieved by seeing to it that the intensity readouts from SS1 are as large as possible. The two sensors situated closest to SS1 (SS2 and SS4 in Fig. 3) serve to enable ‘stereo vision’, i.e. they provide information about in which direction to spin the RxSU in order to achieve maximum intensity at SS1. For example, if the intensity difference $I_{SS4} - I_{SS2}$ is positive (by some margin where noise is taken into account), then the FFU needs to spin clockwise (with a negative value of $\theta$), and vice versa.

The fourth sun sensor (SS3 in Fig. 3) compensates for albedo effects from the Earth, which are due to the sunlight being reflected in its surface. These are typically around 35% of the incident sun intensity and cannot be neglected [5]. Should SS3 not be present, then the sun sensor system might erroneously lock onto the ‘light source’ stemming from the albedo effect because it is unaware of there being a stronger light source present, namely the sun.

When deciding where to mount the four sensors, one needs to – in addition to the fact that they should be evenly spaced – take into account what is considered the optimal position of the sun in relation to the IR sensor aperture. Because SS1 is, ideally, always facing the sun, then there exists a polar angle, $\theta_a$, between the aperture and SS1, which corresponds to the optimal aperture–to–sun angle that can be predetermined. One part of determining this angle is knowing in which direction the FFUs are ejected from the REXUS rocket. This is done by mounting yet another sun sensing system onto the RMU, which keeps track of the RMU–to–sun relation and ejects the FFUs once this is optimal. Using this method, parameters such as the time of day become irrelevant, because the only thing that matters is the angle of the sun within the reference frames of the ISAAC–Rx and the RMU, respectively.

2) Distance Dependency of LED Intensity: The intensity of the LEDs will, of course, not be the same at any distance between the FFUs. In fact, the intensity of one LED will be 'spread out' over a spherically shaped surface according to

$$I(r_d) = \frac{\Phi_0}{\int_0 r_d^2 \sin \theta d\theta d\phi} \sin \theta = \frac{900 \text{ mW}}{\int_0 r_d^2 \sin \theta d\theta d\phi} \approx \frac{304 \text{ mW/m}^2}{r_d^2} \quad (1)$$

where $\Phi_0$ [W] is the radiant flux of the LED, $\Omega$ is the total solid angle across which the light radiates, $r_d$ is the separation distance between the FFUs, and therefore the integral sum in the denominator equals the area of the surface over which the light spreads at $r_d$.

In the particular case of the LEDs used for the ISAAC project, $\Phi_0 = 900 \text{ mW}$ per LED and $d\Omega = \sin \theta d\theta d\phi$ with $\theta : -\theta_v \rightarrow \theta_v$ and $\phi : -\theta_v \rightarrow \theta_v$, where $\theta_v = 65^\circ$ is the angular interval from the center axis within which 90% of the total flux is radiated [6]. This yields:

$$I(r_d) = \frac{900 \text{ mW}}{\int_0 r_d^2 \sin \theta d\theta d\phi} \approx \frac{304 \text{ mW/m}^2}{r_d^2} \quad (2)$$

The intensity at the aperture then equals $I(r_d)$ multiplied by the area of the aperture.

3) Angular Rate Sensor: The angular rate sensor system is implemented using heritage from MUSCAT, a previous REXUS team [7]. It utilizes the Coriolis effect to measure angular rates. Coriolis forces on a rotating rigid body like the RxSU are on the following form [8]:

$$\mathbf{F}_{\text{cor}} = -2m\omega \times \mathbf{v} \quad (3)$$

where $m$ is the mass of the object and $\omega$ is the angular rate of the object. Hence, $\mathbf{F}_{\text{cor}}$ is proportional to the angular rate of the object. The angular rate sensor consists of a MEMS tuning fork which, upon being subjected to Coriolis forces (the most dominant ones in a rotating system such as the RxSU), vibrates at its eigenfrequency with a velocity, $v$, of

$$v = a \sin \omega_0 t \quad (4)$$

where $\omega_0$ is the eigenfrequency of the tuning fork, and $a$ is the amplitude. Then, using (3), (4) and the fact that $\omega \perp \mathbf{v}$ in the case of the tuning fork on the FFU, the Coriolis force becomes

$$\mathbf{F}_{\text{cor}} = -2ma\omega \sin \omega_0 t \quad (5)$$

The tuning fork is deformed, periodically and elastically, by the Coriolis force, thus producing a current, $Q$, which can be measured electronically [9]. It takes the form

$$Q = A\omega \sin \omega_0 t \quad (6)$$

where $A$ is an amplitude constant, dependent on the material and the geometry. By knowing $A$, $\omega$ can be found.
C. Algorithm Concept

1) Finding hotspots: A crucial part of the tracking algorithm is being able to find the LEDs in the images captured by the tracking camera. Noise is expected and makes it more difficult to find the LEDs. In order to narrow down the image to a set of possible areas, there needs to exist an algorithm which locates hotspots – bright areas that represent prospective locations of the LEDs. Firstly, one needs to define a threshold. An area containing pixels with brightness above this threshold will be considered a hotspot. The challenge is to determine an appropriate value of this threshold brightness – the fewer presumptive LED locations the better, but selecting an all too high threshold increases the risk of not finding any LEDs at all.

The algorithm begins by receiving pixel information from one captured image and every pixel is gone through and is compared to the intensity threshold. It is also checked whether or not the current pixel already belongs to a previously encountered hotspot. If the pixel intensity is above the threshold and does not belong to a previously encountered hotspot, then a new hotspot is considered to be found and the pixel is added to it. When a new hotspot has been defined, the algorithm begins expanding the hotspot by recursively assessing whether the neighboring pixels reach above the threshold. If a neighbor does qualify, that pixel is also added to the hotspot, after which all of its neighbors are checked as well. This continues until all pixels belonging to the hotspot are found. When all neighboring pixels in the hotspots have been found, the algorithm continues with the next pixel along the row which does not belong to the hotspot and starts looking for the next hotspot as described earlier. When the image is completely searched through, a number of hotspots has been found.

2) Tracking modes: In order to be able to perform the tracking described in Section II-A a control algorithm is needed. This algorithm takes a sun angle, an angular rate and a list of possible hotspots as input. The output is the desired mirror angle and the despin velocity. This algorithm consists of three modes referred to as the Despin, Lock and Search modes.

When the ISAAC–Rx is ejected from the rocket it will spin with an angular rate of 4 Hz. At this stage, the algorithm is in the Despin mode. In this mode the main inputs used are the sun angle and the angular rate. The goal is to stop the RxSU from spinning and set the angle to the sun in such a way that the aperture is focusing on ISAAC–Tx (which was described in Section II-B1). In total, this corresponds to

\[ \dot{\theta} \rightarrow 0 \land \theta \rightarrow \theta_{desired}, \]

where \( \theta \) is the angle between SS1 (in Fig. 3) and the sun. When these requirements are met, within a defined error margin, the algorithm proceeds to Lock mode.

In Lock mode, the algorithm starts receiving input from the camera images, which have been previously pre-processed according to Section II-C1. This information consists of a list of possible hotspots in which the LEDs might be found. In the first iteration, it presumes that the most likely hotspot in the first image (the brightest one) actually are the LEDs. In the following iterations it uses information from earlier hotspots to better predict where the LEDs are. It maps hotspots from previously taken images to the current one and is therefore able to track the movement of individual hotspots. These movements are compared to the input movement from the sensors which makes it possible to discard some hotspots as noise. If the angular rate increases so much that the LEDs might get out of the field of view or if the sun angle error becomes too large, it means that ISAAC–Rx has begun to rotate out of the optimal frame-of-view and the algorithm enters Despin mode once again.

If in Lock mode one does not find any hotspots at all, it most likely means that the LEDs are not in the field of view of the camera. This triggers the Search mode. In Search mode, the ISAAC–Rx is commanded to start a slow rotation in which it still receives input from the camera. It then begins searching for possible hotspots, and as soon as one has been found (i.e. when a hotspot has been encountered and the condition (7) is fulfilled) it again enters Lock mode.

The complete concept is shown in Fig. 4.

![Fig. 4: Conceptual diagram of the tracking modes](image)

III. ELECTRONICS OVERVIEW

A. FPGA

FPGAs are integrated circuits with programmable logic which are able to direct and process digital signals. Voltages at the terminal pads of the FPGA are related to some given interval (commonly between 0 and 3.3 V). If the voltage is below the midpoint of this interval, then the signal is considered LOW (i.e. a digital ‘0’ bit), and if it is above the midpoint it is considered HIGH (i.e. a digital ’1’ bit). Digital information, consisting of sequences of these bits, is transferred in between different instances of the FPGA and/or different devices in relation to one or several clocks which are often generated by physical oscillators with known frequencies. The signals from this oscillator can be divided into lower frequencies, should this be necessary. Table I shows how the oscillator frequency is divided on the tracking PCB.

<table>
<thead>
<tr>
<th>System Clock Vector</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M TIME[0]</td>
<td>32.768 MHz Oscillator frequency</td>
</tr>
<tr>
<td>M TIME[1]</td>
<td>16.384 MHz</td>
</tr>
<tr>
<td>M TIME[2]</td>
<td>8.192 MHz</td>
</tr>
<tr>
<td>M TIME[3]</td>
<td>4.096 MHz</td>
</tr>
<tr>
<td>M TIME[4]</td>
<td>2.048 MHz</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>M TIME[25]</td>
<td>1 Hz</td>
</tr>
</tbody>
</table>

TABLE I: Clock reference vector [10]
1) FPGA Programming: For the ISAAC tracking system, ProASIC3 FPGAs from Microsemi are used. The particular model chosen, the A3P250, has 250,000 logic gates and 68 user-customizable I/O ports [11]. Logic is programmed in the VHDL programming language which is written in the Libero IDE suite provided by Actel. The code is designed using a set of modules: a unit class within the code where the firmware designer defines input and output ports and signals which then are routed in relation to the clock. Signals are handled both internally within the module and sent in-between modules. When the VHDL code has been written, the developing environment synthesizes a file containing the logic gate setup which is to be used on the FPGA. In the last step, a JTAG interface is used between the computer and the FPGA, which then can be programmed according to the synthesized file.

2) I2C Communication: In order to enable the tracking PCB to communicate digitally with the different sensors, I2C is used. It is a serial data interface which is used to handle communication between digital circuits. It consists of two signals, the Serial Clock (SCL) and the Serial Data (SDA). One of the devices (the FPGA chips in the case of the ISAAC tracking system) is said to be the master, which is responsible for generating the SCL and initiating communication with the slaves. These receive the SCL and respond to signals sent by the master. Data are transferred through the SDA by relating to the square wave shape of the SCL. Events, i.e. sequences of bits transferred between the devices, occur according to pre-defined firmware routines in which SCL and SDA depend on each other. The I2C interface proves useful when, for example, writing to and reading from the registers of a digital circuit. Registers store bits of information pertaining to the functionality of the device (such as write speeds, hardware configurations, etc.). (Refer to Section III-B4 for the configuration of registers in the case of the CMOS camera.)

B. Tracking Camera

The MT9T031 sensor from Aptina is used for the tracking system. It is a 1/2.5 inch CMOS active-pixel digital image sensor and has an active imaging pixel array of 2592 by 1944 pixels [12]. The sensor is able to perform column and row skipping and pixel binning, which enables the user to read images of preferred sizes (at the expense of the resolution). It is also possible to control in detail the ways in which the exposure is carried out. Table II shows the pin configuration of the sensor header board.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Type</th>
<th>Pin description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCLK</td>
<td>IO</td>
<td>I2C Serial Clock</td>
</tr>
<tr>
<td>SDATA</td>
<td>IO</td>
<td>I2C Serial Data</td>
</tr>
<tr>
<td>TRIGGER</td>
<td>Input</td>
<td>Initiates triggering</td>
</tr>
<tr>
<td>STROBE</td>
<td>Output</td>
<td>Initiates data readout</td>
</tr>
<tr>
<td>EXTCLK</td>
<td>Input</td>
<td>Master Clock</td>
</tr>
<tr>
<td>DOUT[11:0]</td>
<td>Output</td>
<td>12 bits of pixel data</td>
</tr>
<tr>
<td>PIXCLK</td>
<td>Output</td>
<td>Pixel readout clock</td>
</tr>
<tr>
<td>LV</td>
<td>Output</td>
<td>Frame Valid</td>
</tr>
<tr>
<td>LV</td>
<td>Output</td>
<td>Line Valid</td>
</tr>
</tbody>
</table>

TABLE II: Pin layout of the camera

1) Blanking: In addition to the active imaging pixels (i.e. the pixels that are part of the actual output image), there are two blanking areas and two dark columns, which serve as offset from the sensor edges. Blanked pixels are indeed exposed when the camera is triggered (refer to Section III-B3), but they are not included when the data are read out. Dark columns are read, but not exposed and function as a black reference for the internal ADC of the sensor. Fig. 5 illustrates this.

2) Skipping and Binning: Skipping of rows and columns is used to make the resolution of the output images smaller without reducing the field of view of the camera. An example of this is shown in Fig. 6 where every second row and column is skipped and is displayed grey in the image. Skipping can, however, introduce undesired aliasing effects, which results in poor, distorted images [13]. These effects can be reduced by performing row and column binning. This method combines pixels over some pre-defined number of adjacent rows and columns. It can be done either by averaging or by summing over the intensities of the pixels. The concept of binning is shown in Fig. 7.

3) Exposure mode: The default exposure mode for the CMOS sensor is to continuously output frames at a constant, configurable frame rate. In order to better control the data flow for the tracking system, non-continuous triggering may be desired. This is made possible in the Snapshot mode, which
allows the user to capture one image at a time, and is set by writing '0100 0001 0000 1111' to the R0x01E register [14]. The sensor has two triggering pins, TRIGGER and STROBE: TRIGGER enables the exposure, and STROBE enables the actual readout of the data from the exposure. These pins are used in essentially the same way for all the available exposure modes [12]:

1) Wait for TRIGGER to occur, then start the exposure
2) Wait for STROBE to occur, then start the readout

It is the time interval and/or the pixel distance between these two triggerings which differs between the exposure modes. In Snapshot mode, the enabling TRIGGER is done manually and initiates the exposure of the first row (with some delay because of the dark column blanking which, as mentioned previously, are non-exposed pixels that can be used to fine tune the black level, but is not part of, the output image). After a small period of time, \( t \), the second row begins exposing. After an additional \( t \), the third row is exposed, and so on. When \( n_{\text{rows}} \cdot \text{ROW} \) has passed (where \( n_{\text{rows}} \) is the number of rows on the array), all rows are being exposed upon which STROBE triggers automatically. This marks the beginning of the data readout sequence. Once the first row has been read out, TRIGGER can be enabled again, thus making it possible to begin another exposure even before the previous image has been fully read out. Figure 8 shows the timing of the Snapshot mode.

4) Register Configuration: Settings like binning and exposure modes are communicated to the camera by setting the sensor registers to desired values. Registers are written to the sensor using the \( I^2C \) protocol. The sensor is the slave and the control FPGA (see section IV-A) is the master.

A write sequence is initiated by the master setting the SDA line LOW while the SCL line is HIGH. This is called a start bit. This start bit is then followed by the specific 8-bit slave address of the CMOS sensor sent on the SDA line with one bit per SCL cycle when SCL is HIGH. During the next clock cycle, the master releases the data line and the slave pulls it LOW in order to communicate that the information was received. This is called an acknowledge bit. Thereafter, the register address is sent to the slave followed by the actual register data which consists of two sequences with 8 bits of data transferred in the same way described above, each followed by another acknowledge bit from the sensor.

An example of this process is given in Fig. 9 (in which the clock line is called SCL and the data line is called SDATA). In the example the R0x09 register is written to with the value '0000 0010 1000 0100'.

Table III shows the registers that are relevant for controlling skipping, binning and exposure modes.

<table>
<thead>
<tr>
<th>Register Address</th>
<th>Register Name</th>
<th>Register Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0x022</td>
<td>Row Address Mode</td>
<td>Controls row binning and skipping</td>
</tr>
<tr>
<td>R0x023</td>
<td>Column Address Mode</td>
<td>Controls column binning and skipping</td>
</tr>
<tr>
<td>R0x01E</td>
<td>Read Mode 1</td>
<td>Controls exposure modes</td>
</tr>
</tbody>
</table>

TABLE III: Particularly relevant camera registers

5) Capturing: Images are captured from the camera sensor in relation to its pixel clock (PIXCLK) output pin: the 12 data bits in each pixel will be parallely captured with every cycle of PIXCLK. The frame valid (FV) and line valid (LV) output pins give information about whether the sensor is transferring valid pixel information or reading from one of the blanking areas. When FV is HIGH, which it will be an integral number of times equal to the number of rows that are read out, it means that the sensor is outputting pixels on rows containing active pixels (but do not necessarily belong to columns within the active image area). Similarly, then LV is HIGH, the pixels that are output are horizontally aligned with the active image pixels. Thus, when both FV and LV are HIGH simultaneously, the sensor outputs pixel data belonging to a valid image [12]. The 12 bits of data in each pixel are synchronized with the pixel clock and are sent in parallel through DOUT (12 pins) with 12 bits every clock cycle which is seen in Fig 10.

The default operational frequency of the CMOS camera is 96 MHz. At the full resolution of 2592x1944 pixels (5 megapixels), the frame rate becomes 14 fps (frames per second). In the ISAAC tracking system, the highest possible master clock frequency of the oscillator being considered for the experiment is 32.768 Hz (refer to Table I), and thus the frame rate at maximum resolution would be approximately 4.7 fps, i.e. just above one image per revolution of the FFUs at their initial spinning rate. However, in order to increase the frame rate, the resolution may be lowered by either performing skipping, binning or reducing of the field of view. For example, at a resolution of 640x480, the frame rate becomes approximately 41 fps [12].

6) Color Filter: The LEDs emit light with a wavelength peak within [655, 670] nm [6]. This means that a vast part of
the spectrum contributes to the noise in the camera images. Therefore, a bandpass color filter is used with its peak at 660 nm and 10 nm bandwidth [15]. It is mounted together with the camera optics on the RxSU.

C. Sun Sensor

The main component of the sun sensing system is a Silonex SLCD-61N2 photodiode. When subjected to incident light, it generates a current proportional to the intensity of the light, in accordance with the photovoltaic effect [16]. This generated current is converted to a voltage change in a load resistor, which is then amplified using the LM7321 operational amplifier from Texas Instruments and digitized in the MAX11617 ADC from Maxim Integrated [17]. The acquired data are then used in the tracking algorithm in every stage of the data gathering process. Figure 11 shows the schematics of one such sun sensor circuit.

The particular photodiode used has a spectral sensitivity of $\lambda_s = 0.55 \text{ A/W}$, and an area of $A = 21.4 \text{ mm}^2$ [18]. Together with the fact that the sun radiates approximately $I_{\text{sun}} = 1300 \text{ W/m}^2$ on Earth, and that the current is converted to a voltage drop in a 100 $\Omega$ resistor, one would expect there to be a maximum voltage change of

$$\Delta U = \lambda_s A I_{\text{sun}} \cdot R_{\text{load}} = 0.55 \cdot 21.4 \cdot 10^{-6} \cdot 1.300 \cdot 100 = 1.53 \text{ V} \quad (8)$$

D. Angular Rate Sensor

The L3G4200D three-axis digital gyroscope from STMicroElectronics is used. It has a customizable measuring range of $\pm 250/\pm 500/\pm 2000$ dps and a selectable bandwidth of 100/200/400/800 Hz. The sampling rate can be set to 100/200/400/800 Hz. The gyroscope is controlled using the I2C interface [19].

IV. IMPLEMENTATION

A. Schematics and Layout

The tracking PCB on the RxSU needs to handle several different interfaces: communication with the three sensors (the CMOS camera header-board and the two ADCs which convert the analog sun and angular rate sensor signals into digital signals), writing data from every instance of the tracking sequence to flash memories, executing image pre-processing and control algorithms and actuation of the motors which control the RxSU rotation and mirror tilt angle. Due to physical limitations in the number of I/O pads on the FPGAs, and because of their limited computational capacity, four FPGA chips are used, configured as follows:

- **FPGA 1a** is responsible for communicating with and collecting data from the CMOS camera. It reads to and writes from the registers in order to change configurations regarding binning, row skipping and the digital gain of the pixels, for example. It also writes each image captured to a flash memory which is saved for post-flight processing. Lastly, it sends the image data to FPGA 2.
- **FPGA 2** executes the pre-control processing of the captured image. Each image has a certain number of areas-of-interest (hotspots) in which the LED source might be
visible. FPGA 2 locates and takes note of these particular areas and sends information about them to FPGA 3. The purpose of this is to reduce the amount of data being handled in FPGA 3. Because this algorithm cannot be run continuously as the pixel data are being received from FPGA 1a, there needs to be a buffer from which FPGA 2 can read once the entire image has been received. This buffer is a RAM which is connected to FPGA 2.

- **FPGA 1b** is responsible for communicating with and collecting data from the sun and angular rate sensors and write it to a flash memory. It also pre-processes the data from the sun sensors as described in section II-B1. Once finished, the angular rate and sun location data are sent to FPGA 3.

- **FPGA 3** processes the hotspot data uses the data from FPGAs 1b and 2 to execute the control algorithm that was described conceptually in section II-C. The results of this are stored onto a flash memory. Also, logical pulses are sent to current-controlled circuits which power the rotation and mirror tilt angle stepper motors.

Fig. 12 shows a conceptual layout of the FPGA configuration of the RxSU tracking PCB.

The schematics of the PCB are drawn in Mentor Graphics, a CAD software. Once finished, the actual layout of the PCB is created, i.e. the specific components and their wiring are routed on the board. The PCB designs are then sent to an external company which manufactures the board, with wiring and solder pads. The components are then soldered by hand at SPP.

![FPGA configuration of the RxSU tracking PCB.](image)

Fig. 12: FPGA configuration of the RxSU tracking PCB. The FPGAs with dashed borders are those executing algorithms, and those with non-dashed borders acquire data.

### B. FPGA Programming

The communication with the camera is handled with two modules called Camera Core and Camera Controller as seen in Fig. 13. The Core is responsible for the direct communication with the camera sensor through the I^2C protocol while the Controller gives information to the Core about what data to send and when to send them. The arrows indicate in which direction each signal is sent internally and Table IV explains them.

Initially, the Core and the Controller are in an idle state in which the serial output pins (SCL and SDA) connected to the camera both are HIGH. The Controller then initiates the communication by setting the CoreReturn signal (viewed upon as trigger) to HIGH. The Controller also sends the slave address (SlaveAddress), the number of bytes to write (NumberOfBytes) and then the first byte to write (through DataOut). The slave address holds information about which component to communicate with and also whether it is a read or write sequence that is to be executed. For example, in the case of a write sequence to the camera sensor, the 0xBA slave address is used [12]. The NumberOfBytes signal indicates how many bytes that is to be written, and because the camera registers consist of two bytes this is set to 0x02 (refer to Section III-B4). The Controller then waits for the Core to communicate with the camera sensor and because it possesses information about which state the Core is in (by the I2CStateIn signal), it knows when the Core is finished. The Controller then sends the second byte to write to the register through DataOut. Because the Core knows that only two bytes are to be written, it writes the last byte and thereupon ends the I^2C communication.

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Signal Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLK</td>
<td>External clock</td>
</tr>
<tr>
<td>Reset</td>
<td>External reset signal</td>
</tr>
<tr>
<td>CoreReturn</td>
<td>Signals to the core when to start.</td>
</tr>
<tr>
<td>SlaveAddress</td>
<td>Gives the slave address of the camera to the core.</td>
</tr>
<tr>
<td>DataIn/DataOut</td>
<td>Gives data to send to the camera to the core.</td>
</tr>
<tr>
<td>NumberOfBytes</td>
<td>Sets the number of bytes which are to be written or read by the core.</td>
</tr>
<tr>
<td>StateOut/I2CStateIn</td>
<td>Information from the core about in which state it is.</td>
</tr>
<tr>
<td>SCL</td>
<td>Serial clock to the camera.</td>
</tr>
<tr>
<td>SDA</td>
<td>Serial data to the camera.</td>
</tr>
</tbody>
</table>

**TABLE IV:** Camera Core and Controller signals

### C. Sun Sensor

Using the I^2C interface, data are read out from the MAX11617 ADC to FPGA 1b. The ADC has 10 independent input pins, of which four are used for the photodiode circuits while the others are not connected. All sensors are sampled simultaneously, upon which the data are sent into the control system which will make attempts to align the RxSU with the sun in ways described in Section II-B1.

### D. Angular Rate Sensor

Again, using the I^2C interface, data are read out from the gyroscope and into FPGA 1b. Once the sensor is powered on, settings are configured on the sensor by the FPGA. Then the gyro begins sampling data at a rate which can be configured in
E3. TRACKING SYSTEM FOR ISAAC

![Camera Core and Controller](Image)

**Fig. 13: Camera Core and Controller**

a register in the sensor. The FPGA (which is the I^2^C master in this case) enters a loop in which it monitors the state of a status register for a ‘data-ready’ bit. If such a bit is encountered, all the 7 bytes of angular rate data are read out. Appended to the data is also an eighth byte, containing IDs and such. After this, the data are complete and can be written to the flash memory and passed onto FPGA 2 [10].

V. TEST PROCEDURES

Below are descriptions of three tests conducted in order to assess the performance of the different components of the ISAAC tracking system.

A. Sun Sensors

One desires information about the performance of the sun sensing system when subjected to sunlight from different angles, \( \theta \), in the plane of the RxSU. Therefore, a quantitative test was performed. The test was set up to simulate as closely as possible the real scenario of the ISAAC experiment: Four photodiode circuits of the kind described in Fig. 11 were placed on the circumference of a circle with a radius of 120 mm (which matches the dimensions of the ISAAC-Rx) [4]. The sensors were placed evenly spaced according to Fig. 3 – each sensor offset by \( 90^\circ \) from its neighbors. The reference frame was set so that \( \theta \) was zero in the direction of the sun. A screen of plain, white paper was installed at \( \theta = 180^\circ \) and a distance of 500 mm from the center of the circle so as to imitate albedo effects somewhat.

Measurements were carried out by noting the voltage output from each of the sensors at angles \( \theta : 0^\circ \rightarrow 350^\circ \) with increments of \( 10^\circ \).

B. LED Distance

The tracking solution of the RxSU almost entirely depends on how well it is possible to capture images of the LED mounted onto the TxSU. The goal of this test was to give an overview and to investigate at what distances one LED was visible to the camera.

Using a demonstration PC software from Aptina Imaging, which enables simple image capturing through a USB header board, a sequence of images was captured at various distances.

The full pixel-array was read out as opposed to implementing binning and skipping methods discussed in Section III-B2. The PC software was set to convert all pixel intensities to digital signals using the same gain setting. The LED was then located manually in the image, upon which the brightnesses of its pixels were noted, and the brightness of the background was subtracted so as not to disturb the comparativeness between the different images.

C. Hotspot Algorithm

In order to understand how well the concept of finding hotspots in images (described in Section II-C1) functions, a test program was developed. The program was written in C# and implements the concept which is then tested on sample images, both generic ones and images captured from the camera. The program scans through the images and outputs information about which hotspots it has found along with the pixels belonging to them.

Fig. 14 shows an example image that was created using Paint.NET. This image has some bright areas which, depending on the threshold chosen, could be considered to be hotspots.

Fig. 15 shows an image captured with the camera sensor, containing an LED at a distance of 340 m. The white circle indicated the location of this LED.

VI. RESULTS

A. Test Results

1) Sun Sensors: The plot in Fig. 16 illustrates the results of the measurements described in Section V-A:

In this figure, \( \theta \) corresponds to the angle between the sun and SS1 in Fig. 3.
E3. TRACKING SYSTEM FOR ISAAC

Fig. 15: Hotspot tracking test image. The white circle indicates the location of the LED.

Fig. 16: Sun sensor intensity readouts

2) LED Distance: Fig. 17 shows a fitted graph of the data points measured. The fitted curve has the following equation:

\[ f(x) = ax^b \]  \hspace{1cm} (9)

The results of the curve fitting were the following:

\[ \begin{cases} 
  a = 7.363 \cdot 10^9 \\
  b = -2.851
\end{cases} \]  \hspace{1cm} (10)

3) Hotspot Algorithm: Table V shows how the number of hotspots found in each image (Fig. 14 and Fig. 15) depends on the threshold value choosing and using the test program described in Section V-C. The threshold value ranges from 0 to 1 where 0 corresponds to black and 1 to white.

<table>
<thead>
<tr>
<th>Image</th>
<th>Threshold value</th>
<th>Hotspots found</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic (Fig. 14)</td>
<td>0.8</td>
<td>3 (all white)</td>
</tr>
<tr>
<td></td>
<td>0.7</td>
<td>5 (2 white, 2 grey, 1 mixed)</td>
</tr>
<tr>
<td>From camera (Fig. 15)</td>
<td>0.9</td>
<td>1 (the LED)</td>
</tr>
<tr>
<td></td>
<td>0.85</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>0.8</td>
<td>271</td>
</tr>
</tbody>
</table>

TABLE V: Number of hotspots found depending on threshold value

B. Project Outcome

The following sections outline the final outcome of this bachelor thesis project, in terms of the results.

1) Hardware: As for the hardware, the goal of the thesis project was to construct a sun sensing system and a tracking PCB that could handle the incoming data from the sensors. Also, the two existing sensor systems, namely the angular rate sensor and the CMOS camera, were to be tested and incorporated into the tracking system.

The sun sensor system has been constructed and its functionality has been tested (refer to Section VII-A1 for the discussion of the results). A test PCB was designed according to Fig. 11 and has been used throughout the course of the project. A more space-efficient PCB will be designed to fit better on the skin of the RxSU.

As for the tracking PCB, the schematics have been designed as specified in Section IV-A. Each component has been chosen and all the schematic wiring has been routed. However, the actual layout has not been drawn, and because of this, no PCB has been ordered.

The angular rate sensor coming from the MUSCAT team has been studied on a conceptual and theoretical level. Its incorporation into the control algorithm has been developed. However, it has not actually been tested for the purposes of the ISAAC tracking system.

2) Software and Firmware: Included in scope of the software development part of the thesis project was to develop the signal communication between the onboard sensors and the FPGAs. Also, a tracking algorithm was to be designed to control the tracking.

The software that has been developed is:

- The I²C register communication with the camera sensor which controls settings like binning, skipping, exposure modes and such. (Refer to Section III-B.)
- Storage of image data from the camera sensor to the on-chip FPGA memory buffer.
- Reading data from the on-chip FPGA memory buffer and sending it through a COM interface for viewing on a connected computer.
- A test program to demonstrate the concept of finding hotspots.
A conceptual design of the control algorithm has been developed.

The software that has yet to be developed is:

- An optimized hotspot finding algorithm written in C that can be executed by an embedded core FPGA.
- Data gathering procedures for the sun sensors and the angular rate sensor.
- The implementation of the tracking algorithm as described by the concept in Section II-C2.

VII. DISCUSSION

A. Test Results and Performance

1) Sun Sensor: As stated in Section II-B1, the intensity readout should assume a periodic behavior according to Fig. 18.

![Fig. 18: Theoretical sun sensor voltage readouts](image)

Again, $\theta$ is the angle between the sun and SS1 as described in Fig. 3. The graph shows the intensity readouts in the case of no albedo effects present. The calculations were carried out using the circuit design showed in Fig. 3 and eq. (8).

As can be seen in this figure, one expects readouts between 0.4 V and 2.9 V. The lower limit was chosen so as not to get signals too close to the supply rail of the operational amplifier, because such signals would not be amplified at all. This rail distance is amplifier-specific – 0.3 V in the case of the LM7321 amplifier which was used for the sun sensors. However, the limit to the upper rail was underestimated: it should also have been at least 0.3 V as specified by the LM7321 data sheet. In the setup used for the test, the rail-to-rail distance was 3.2 V, with ground being the lower limit. Therefore, with a maximum output of 2.9 V, one would expect the amplifier to clip somewhat. This phenomenon can indeed be seen in Fig. 16, where the signal is clipping at just about 2.9 V. This means that the signal would have reached a peak higher than the figure shows, if the setup would have been made in a more correct manner. The results are therefore quite poor in that sense (but not completely useless in others as can be seen below). They are, however, easily corrected, time allowing, by either increasing the supply voltage of the operational amplifier or lowering the feedback gain.

From Fig. 11, one expects there to be a ‘gap’ between the intensity peaks in which the intensity maximum of the sun falls precisely in-between two sensors. This would present problems with data processing because of the closeness to the noise (i.e. albedo) floor in the ‘blind spots’. This appears to be confirmed by Fig. 16 – if the signal would not have been saturated, then it seems that the gaps would have been present as per the theoretical supposition.

One way of circumventing this problem would be to use a larger amount of photodiodes (at the expense of geometrical space on the FFU as well as increased complexity in the processing) so as to narrow these gaps.

In the other end of the readouts in Fig. 16, the albedo effects do indeed affect the results at low sun intensities by increasing the noise floor. This was expected but do seem compensable by the relatively strong signals at greater intensities.

2) LED Distance: The curve in Fig. 17 predicts visibility at a distance of well over 500 meters (depending, of course, on how one chooses to define an LED being visible, in terms of numbers of bright pixels and such). However, the method is rough and effort has to be put into constructing an algorithm which detects the LED in a more accurate manner. Furthermore, weather and other measuring conditions were not optimal at the occasion – sunlight fell upon the LEDs in such a way that the brightness of the background was unnecessarily elevated (which it would not be in the real experiment due to the sun sensor system and the direction of ejection). Also, the camera lens did not perform very well in focusing on the LED at the furthermost distances which, too, affects the results. The latter is solved in the ISAAC experiment by using a different lens.

Geographical conditions did not allow for measurements at further distances than those shown in Fig. 17. Such measurements would have provided useful information for the tracking solution, because the maximum operating distance between the FFUs has yet to be definitively decided upon by the ISAAC team. Should the spectroscopy require a larger distance than 500 meters, then the only indicator of such a distance being suitable for the tracking is the (perhaps somewhat poorly) fitted curve in Fig. 17.

3) Hotspot Algorithm: The testing of the generic image (Fig. 14) shows that the developed program, following the concept described in Section II-C1, is indeed able to successfully find all the hotspots in the image (above a certain threshold). It also considers differently bright, but geometrically connected areas that are above the threshold to belong to the same hotspot (the grey and the white) as desired.

The same result is achieved when testing the image from the camera (Fig. 15). If the threshold is high enough, only the hotspot corresponding to the LED is found. If the threshold is set to a lower value, it also considers other bright areas to be hotspots.
When using the concept of this algorithm in the real tracking system (according to the theory in Section II-A2) there are a number of problems to address. Since the computational power will be limited (refer to Section IV-A) and the localization of the LEDs has to be quick, the algorithm may have to be optimized. It also needs to be rewritten (in another programming language than C#) and compiled in order for it to be able to be executed on an FPGA.

Furthermore, the threshold cannot realistically be set so as to only find the hotspot which corresponds to the actual LED, because the risk of not finding any LED would be considerable. A more sober way of processing the hotspots would be to set the threshold slightly lower, thus finding more hotspots which can be processed by the control algorithm as described in Section II-C.

B. Project Outcome

1) Hardware: As discussed in Section VII-A1, the test conducted concerning the sun sensing system was perhaps not as conclusive as one desires. This does indeed affect the outcome of the hardware implementation of said system, because it has not yet been definitively determined whether the four photodiodes that were decided upon will suffice (even though, as mentioned, the results are reassuring).

The tracking PCB was, as mentioned, not completely designed as far as the actual layout. This is, of course, a failure in the perspective of the thesis project. However, one must take into account the fact that the tracking system constitutes a small part of ISAAC at whole. When it comes to the PCB layouts, the Mechanical team at ISAAC had yet to specify the amount of space available on the RxSU for the tracking, power and IR spectroscopy PCBs. This lead to the working progress of the Electrical team coming to a halt at the point of designing layouts. The schematics is the only part which has been completed.

What remains to be done with the hardware goals stated in Section VI-B1 is mainly the hardware characteristics of the angular rate sensor. One desires to determine its responsiveness to changes in rotation speeds as well as its accuracy in its readouts. Another important thing to determine, which serves as an interface between hardware and software, is the sampling rate at which the sensor collects angular rate data. This would be important in several aspects of the control algorithm described in Section II-C.

2) Software and Firmware: The part of the software development scope concerning the signal communication between onboard sensors has proceeded quite well, both in the sense that the camera can be controlled by setting registers and that data has been successfully read out from the pixel array to an external computer. As for the data gathering from the sun and angular sensors which has yet to be developed, much of the heritage code from previous projects can be used. This is why less effort has been put into this work, contrary to the camera sensor, which is a new kind of hardware for the REXUS teams at KTH.

The tracking algorithms – the hotspot locating as well as the control algorithm – are still in the concept stage. The hotspot algorithm is written in C# for the x86 architecture. However, it would have been preferable to write it directly in C and then test it on the actual FPGA hardware. There are two reasons why this has not been done. Firstly, the time needed to develop the software and then test it would have been out of range for the project. Therefore it was decided to develop it with C# since it provides a much shorter development time and enabled proof-of-concept of the functionality. Secondly, executing C code on an FPGA requires an embedded core, which has yet to be implemented by ISAAC.

The main reason that the algorithms were not finished was that the E3 group did not prepossess enough expertise and experience in the beginning of the project. Much time was spent learning about hardware configuration and design as well as the development tools and the programming languages needed for configuring the software.

VIII. Conclusions

The goal of the ISAAC project has been briefly explained. The specifics about the ISAAC tracking system has been presented; the functionality of the different subsystems, as well as the concepts of the tracking algorithms have been described. The electronics of each subsystem has been described in further detail. Choices of hardware and electronic implementation has been outlined. Strong emphasis has been put on describing the CMOS camera, as far as its electronic characteristics and its signal interfaces. The actual implementation has been described in the sense of layouts and data communication.

Tests concerning the sun sensing system, the distance dependency of LED intensity, and image processing have been presented. Despite being a little problematic in their execution in some cases, they serve as proofs-of-concept for three important subsystems in the tracking system.

In the end of the project, it remains to put together a fully functioning tracking system. However, the different subsystems used for the tracking system have been thoroughly explored. The hardware implementation needs finalizing, and the control algorithm needs coding and testing.

Acknowledgements

We would like to thank our supervisor, Dr. Nickolay Ivchenko, Associate Professor at the Department of Space and Plasma Physics at KTH, for his invaluable support and knowledge. Also, our warmest thanks go to the rest of the students in the ISAAC team, together with whom we have struggled so hard to make a reality of our dreams of space (or, at least, of the middle atmosphere).

References