

# Design of an Image Processing Application Using Off-the-shelf IP Cores.

By Eduardo Bezerra

## 1. Introduction

The purpose of this report is to describe the functionality of the Processing Unit Board (PUB) of the Panopticon project. In order to provide a better understanding of PUB, the basic components of Panopticon are briefly described. Panopticon has two main components, the Base Station and the Remote Station, which is also known as Camera-On-Pole (COP). PUB is one of the three main modules of COP.

This report is divided into three distinct parts. The first part of the report, Section 2, is an overview of the Panopticon system with emphasis on PUB, where block diagrams and UML components are used to explain the functionality of the system as a whole. The second part consists of Section 3, where background information on image processing is introduced. The last part of the report, Section 4, discusses the functional description of PUB, and the interfaces between PUB and the other two components of COP, the Wireless Interface (WI) and the Image Capture Unit (ICU).

## 2. System Overview

The block diagram in Figure 1 outlines the major areas of the prototype system, and it also serves to define the scope of the development work. Three identical remote stations (COP) shall communicate with a base station, although the maximum number of COPs will be defined according to available wireless bandwidth, environmental conditions of each installation, and the compression ratio/image quality required. The diagram in Figure 1 is a high level representation of the required processes. The processes shown in the diagram reflect the general operations required of the system, and are not intended to represent individual components. The system processes and stores compressed images (stream video) on the base station computer.

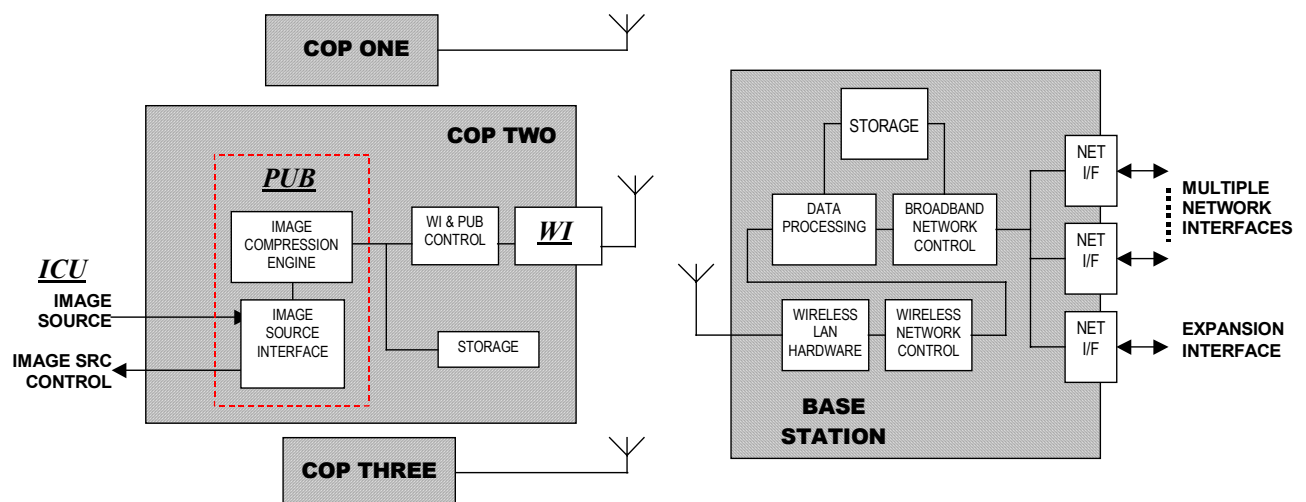


Figure 1. Block diagram of the system to be implemented by the development team.

There are COPs capturing & compressing the images and transmitting them to the base station computer. The system helps the final user:

- to search and to retrieve images (stream video),
- to show the live images that are being captured from a camera,
- to control camera movements,
- to configure the system for the final user
- to present information about system health monitoring.

COP consists of an image input (ICU), a compression engine (PUB) and a network output (WI). It uses a real-time embedded architecture to capture, process and transmit images to the base station via the wireless network medium. A number of COPs are attached to each base station. The base station receives commands from the user to control the COPs, which are then forwarded to the COP(s). The image processing is dynamically controlled based on these user commands. There are system constraints related to COP performance, COP image frame rate and COP processing rate.

The rest of this report discusses aspects related to the design of the PUB module (dashed box in Figure 1).

### 3. Background Information

This Section presents a brief introduction to the concepts and nomenclature used in the image processing module of this project. It is assumed that ICU captures the image (one frame), and sends it as a serial stream of bytes to PUB. The protocol between ICU and PUB is relatively simple. A Vertical Synchronism (V-Synch) pulse is generated by ICU to sign to PUB the start of a new frame. A Horizontal Synchronism (H-Synch) pulse is generated by ICU for each new line of pixels in the frame.

The image generated by ICU can be composed by two kinds of pixels according to the method of colour coding used, Red-Green-Blue (RGB) or Luminance (Y), Chrominance and Saturation (U-V). In this project an RGB pixel can have two different lengths 24 bits or 12 bits. In a 24-bits pixel, each colour (red, green and blue), are represented by 8 bits (1 byte per colour). In a 12-bits pixel the three colours are 4-bits long each.

Assuming the use of YUV as the output of ICU, the two possible formats considered are YUV 4:2:2 and YUV 4:4:4. In YUV 4:2:2, the pixels are composed according to the diagram in Figure 2. For every four bytes of Y, there are two bytes of U and two bytes of V.

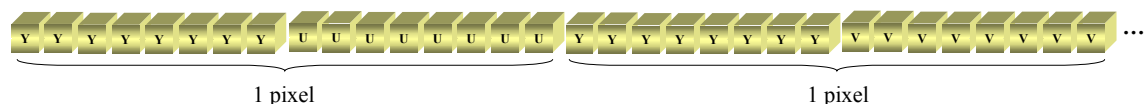


Figure 2. YUV 4:2:2 colour coding.

Figure 3 shows the pixel format for YUV 4:4:4. In this case the three components are organized in a similar way to RGB, i.e. a pixel is composed by three bytes, one for Y one for U and one for V.

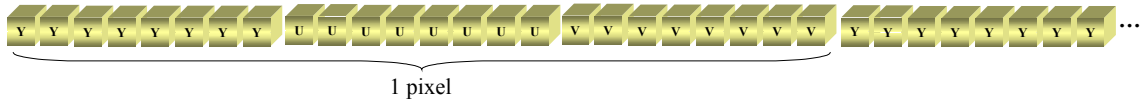


Figure 3. YUV 4:4:4 colour coding.

If the input is in the YUV 4:2:2 format, then the colour components need to be interpolated in order to obtain the format used by PUB, which is YUV 4:4:4. The interpolation is accomplished according to the scheme shown in Figure 4. Considering a YUV 4:2:2 input frame with a size of 256 x 256 pixels, the dimensions of the three components are Y = 256 x 256 pixels, U = 128 x 128 pixels and V = 128 x 128 pixels. The interpolation is done by averaging the missing component (U or V) of the YU and YV pairs. After the interpolation all three components will have the same size: 256 x 256.

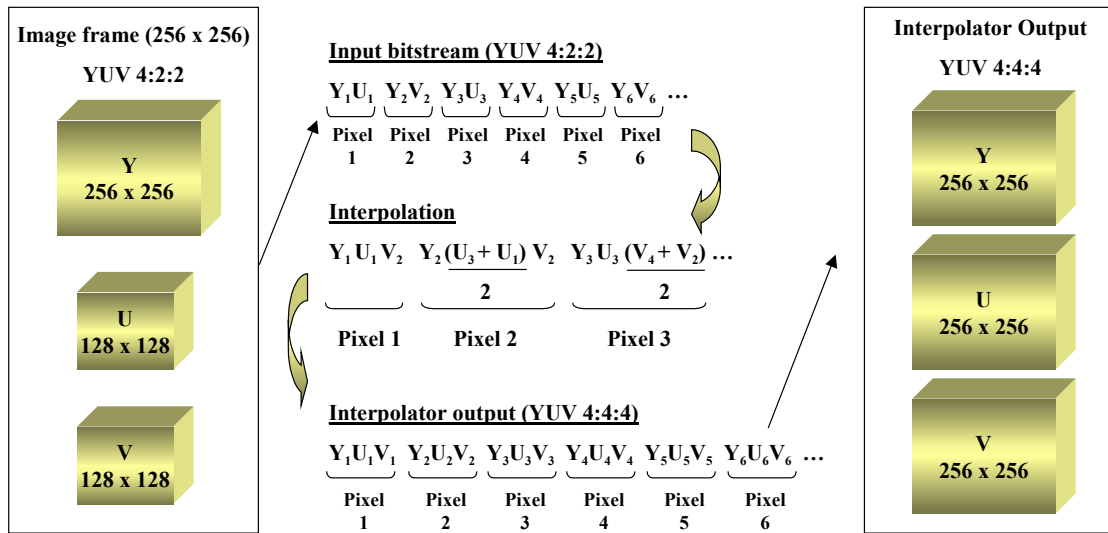


Figure 4. YUV 4:2:2 to YUV 4:4:4 interpolation scheme.

As stated before, ICU can provide also RGB images. In order to convert those images into YUV 4:4:4, the following equations shall be used:

$$Y = \frac{R + 2 * G + B}{4} \qquad U = R - G \qquad V = B - G$$

Another colour format used in this project is Grey-Scale. In this case only one byte is necessary to represent 256 levels of black and white.

#### 4. PUB Specification and Design

The main hardware component of PUB is a Field Programmable Gate-Array (FPGA). The functionality of the FPGA is to be designed and implemented in a high-level of abstraction, using a Hardware Description Language (HDL), and Intellectual Property (IP) Cores. The objective is to produce a design independent on the target FPGA device, or as much independent as possible. Unfortunately, due to Electronic

Design Automation (EDA) tools limitations, it is not possible yet to obtain an efficient design independent of the synthesis tool used. Because the choice of the EDA tools will play a major implication on the design and optimisation of the HDL code, special attention has been given to the selection of the right tools.

An important component of PUB is a commercial IP core that performs the most critical operations of the JPEG 2000 algorithm. In order for this core to be used, the input image needs to be pre-processed, and the whole JPEG2000 encoding operation must be synchronised under the supervision of several controllers. Figure 5 shows the flow of transformations on the image during the encoding process. A Master Controller is responsible for driving the encoding flow, and also for the generation of status information to WI, and control commands to ICU.

The Master Controller, the encoding units, and the interactions between them are described next.

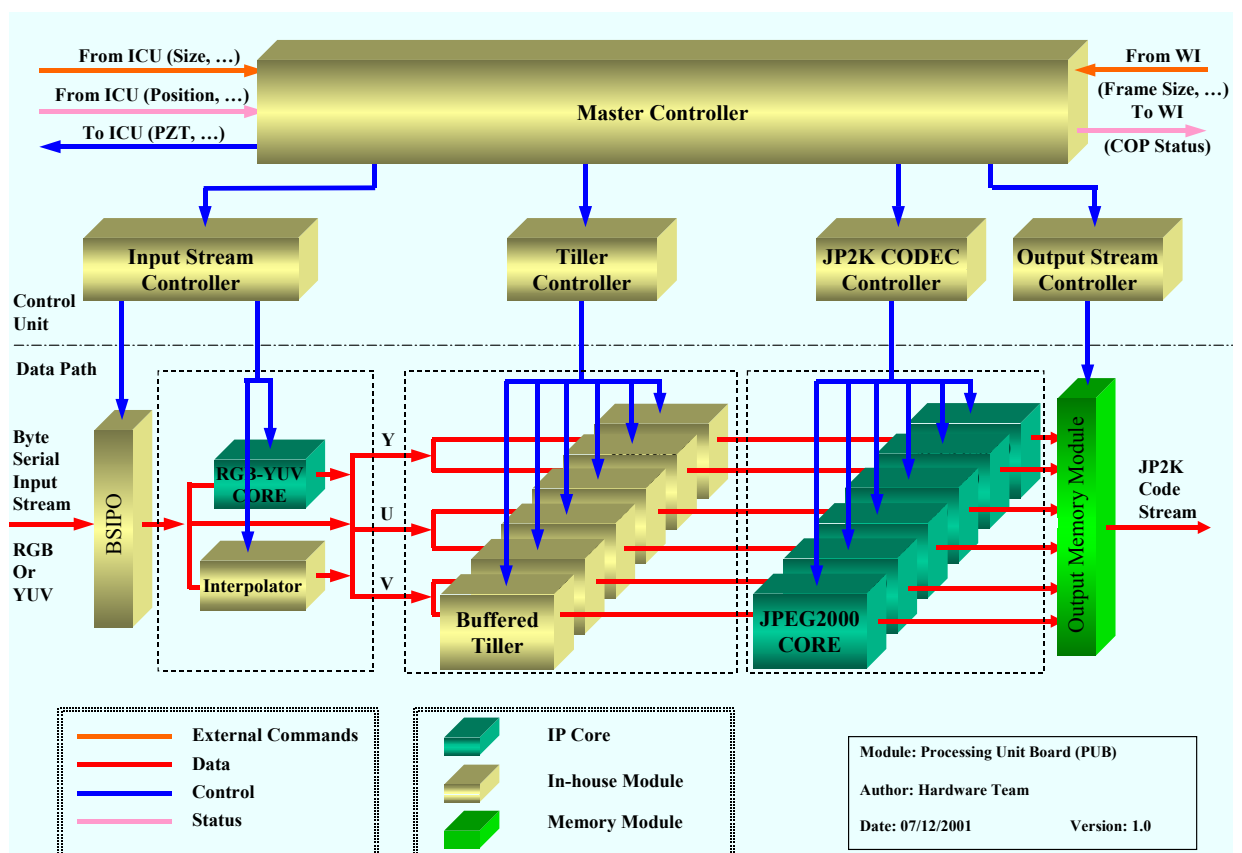


Figure 5. Block diagram of PUB.

#### 4.1. Interfacing PUB, ICU and WI

The interface between PUB and ICU/WI is described in the use case diagram shown in Figure 6. This diagram helps to visualise the source and destination of commands, data and status in PUB. From the diagram it is possible to observe that status information from ICU and position commands from WI, are received by the Master Controller in the FPGA (see Figure 5), but they are not used to drive the encoding flow. The interface between PUB and ICU/WI is detailed in the following tables.

Table I. ICU to PUB interface.

Input stream (image)	<p>The image shall be sent by ICU one byte at a time, and the bytes are processed by PUB in order to obtain the original pixels. The following pixel formats are accepted as input:</p> <ul style="list-style-type: none"> <li>• RGB, 24 bits - a pixel is composed by three bytes representing the three colours;</li> <li>• RGB, 12 bits - a pixel is composed by three nibbles representing the three colours;</li> <li>• YUV, 4:2:2 - a pixel is composed by a series of YU and YV components, where each Y, U and V component is eight bits long;</li> <li>• YUV, 4:4:4 - a pixel is composed by a series of YUV already interpolated components, where each Y, U and V component is eight bits long;</li> <li>• Grey-scale – a pixel is composed by a single byte.</li> </ul>
V-Synch pulses	ICU generates a pulse for each new frame.
H-Synch pulses	ICU generates a pulse for each new row in a frame.

Table II. WI to ICU (through PUB) interface.

Camera positioning (PZT)	As only PUB has a direct interface to ICU, WI receives PZT positioning information from the Base Station and sends it to PUB. The Master Controller in PUB receives the PZT information, and sends it straight away to ICU.
Frame Size	ICU uses this information to set the capture device for the required frame size.

Table III. ICU to WI (through PUB) interface.

Camera position information	When the Base Station requires positioning information from COP, ICU sends the information to WI through the Master Controller in PUB.
-----------------------------	--

Table IV. WI to PUB interface.

Colour Type	<p>Colour type information is used by the Master Controller in the Input Stream Unit. There are three possible colour types:</p> <ul style="list-style-type: none"> <li>• RGB</li> <li>• YUV</li> <li>• Grey Scale</li> </ul>
Colour Style	<p>Colour style information is used by the Master Controller in the Input Stream Unit. The possible colour styles are:</p> <ul style="list-style-type: none"> <li>• 12 bits RGB</li> <li>• 24 bits RGB</li> <li>• 4:2:2 YUV</li> <li>• 4:4:4 YUV</li> <li>• 8 bits Grey Scale</li> </ul> <p>It is important to notice that only two bits might be necessary to select the Colour Style:</p> <ul style="list-style-type: none"> <li>• 1 bit to select between 24 bits RGB or 12 bits RGB</li> <li>• 1 bit to select between 4:2:2 YUV or 4:4:4 YUV</li> </ul> <p>If the Colour Type is Grey Scale, then there is no need to inform the Colour Style.</p>
Frame Size	<p>Frame Size information is used by the Master Controller in the Tiller Unit. The accepted frame sizes are:</p> <ul style="list-style-type: none"> <li>• 128 x 128</li> <li>• 256 x 256</li> <li>• 512 x 512</li> <li>• 512 x 768</li> <li>• 1024 x 1024</li> <li>• 1024 x 1280</li> </ul>
Frame Speed	<p>Frame Speed information is used by the Master Controller in the Jpeg 2000 CODEC Unit. The frame compression ratio is calculated as a function of the frame speed (see Section 4.5). If the image quality is not adequate, then the user can decide for selecting a lower frame compression ratio, by changing the Frame Speed <u>OR</u> the Frame Compression Ratio. As the variable Frame Speed is a function of the Frame Compression Ratio, then just one of them can be used at a time.</p>
Frame Compression Rate	<p>Frame Compression Ratio information is used by the Master Controller in the Jpeg 2000 CODEC Unit. The frame speed is calculated as a function of the frame compression ratio (see Section 4.5). If the image quality is not adequate, then the user can decide for selecting a lower frame compression ratio, by changing the Frame Speed <u>OR</u> the Frame Compression Ratio. As the variable Frame Speed is a function of the Frame Compression Ratio, then just one of them can be used at a time.</p>

Table V. PUB to WI interface.

Status	Error and other house keeping information generated by PUB units.
Jpeg 2000 Code Stream	The compressed image generated by PUB is stored in a memory module, external to the FPGA, and accessible from WI through the Output Stream Unit. The Jpeg 2000 image in the memory can be read one byte at a time.

#### 4.2. Master Controller

The master controller, as shown at the top of Figure 5, activates the Input Stream Controller, the Tiller Controller, the JP2K CODEC Controller, and the Output Stream Controller, according to commands and status information received from ICU and WI.

#### 4.3. Input Stream Unit

The Input Stream Unit, as shown in Figure 5, is composed by the Input Stream Controller, by the Byte Serial Input to Parallel Output (BSIPO), by the Interpolator, and by the RGB-YUV Core.

The BSIPO module under supervision of the Input Stream Controller, uses the colour information (type and style) sent by WI to parallelise the Input Stream of bytes, reconstructing the original pixels sent by ICU. Table VI describes the output of BSIPO, according to the colour information in use.

Table VI. BSIPO operation.

Colour Type	Colour Style	Operation
RGB	24 bits	Red (8 bits), Green (8 bits), Blue (8 bits) – no processing
RGB	12 bits	Red (4 bits), Green (4 bits), Blue (4 bits) – add 4 bits 0 before it colour, e.g. 0000RRRR 0000GGGG 0000 BBBB
YUV	4:2:2	Y (8 bits), U (8 bits), Y (8 bits), V (8 bits), ... - add 8 bits 0 before it YU or YV pair, e.g. 00000000YU 00000000YV
YUV	4:4:4	Y (8 bits), U (8 bits), V (8 bits) – no processing
Grey Scale	8 bits	GS (8 bits) – add 16 bits 0 before the byte

The output of the BSIPO module is always a 24-bits long pixel, and it is the input for the Colour Processing Block (CPB), which is a module composed by the Interpolator and the RGB-YUV Core (dashed box on the left side of Figure 5). Table VII lists the information necessary by the Input Stream Controller to select the components of CPB. The output of CPB is a 3-bytes long YUV pixel used as the input for the Tiller Unit.

Table VII. CPB operation.

	RGB 24-bits	RGB 12-bits	YUV 4:2:2	YUV 4:4:4	Grey Scale
RGB-YUV core	√	√			
Interpolator			√		

By-pass				√	√
---------	--	--	--	---	---

**4.4. Tiller Unit**

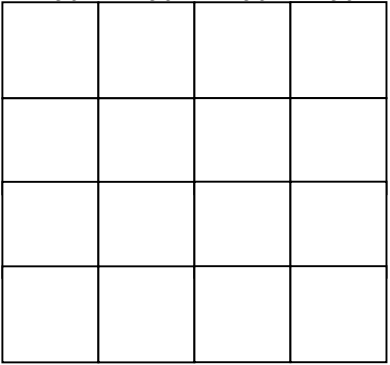
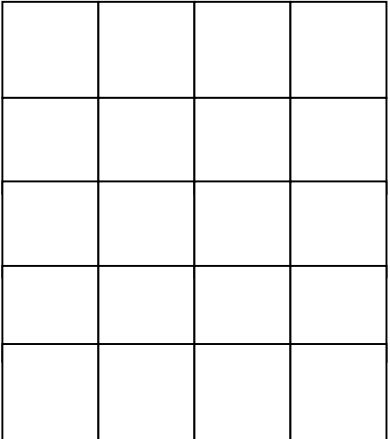
The Tiller Unit is responsible for dividing the image in tiles, and to provide the input to the Jpeg 2000 CODEC Unit. The tiles' dimensions are calculated according to the image size information provided by the Master Controller (from WI). As the Jpeg 2000 core takes as an input tiles as big as 256 x 256 pixels, these dimensions have been chosen as the standard for the tilling procedures. Two strategies for the design of this unit have been investigated. The first strategy makes use of an external memory for buffering the tiles. In the second strategy, the internal memory of the FPGA is partially used. Both strategies deserve deeper investigation, considering different variables as, for instance, the number of instances of the Jpeg 2000 Core, as well as the processing speed of the core.

Table VIII. Frame Tilling for frame sizes smaller than 512 pixels wide.

Frame Size	Tilling Configuration
128 x 128	
256 x 256	
512 x 512	
512 x 768	



Table IX. Frame Tiling for frame sizes greater than 512 pixels wide.

Frame Size	Tiling Configuration
1024 x 1024	<div style="text-align: center;"> <span style="margin-right: 10px;">256</span> <span style="margin-right: 10px;">256</span> <span style="margin-right: 10px;">256</span> <span>256</span> </div> 
1024 x 1280	<div style="text-align: center;"> <span style="margin-right: 10px;">256</span> <span style="margin-right: 10px;">256</span> <span style="margin-right: 10px;">256</span> <span>256</span> </div> 

The algorithm for the tiling process is as follows:

1. Using the Frame Size received from WI, choose a Tiling Configuration from Table VIII;
2. Calculate memory addresses (start/end) to store the tiles, according to Table VIII;
3. Write the tiles to the memory, using the information obtained in step 2;
4. In case of a Grey Scale image, use only one of the three inputs (Y, U or V);
5. Read tiles from memory and send them to the JPEG 2000 core, using the handshake information from the core Datasheet.

For the frame sizes listed on Table VIII, the internal memory of the FPGA can be used for the tiling operation. In this case, it will be necessary to use up to six copies of the JPEG 2000 IP core. For each component (Y, U and V), there will be two JPEG 2000 IP cores. This strategy is shown in Figure 6.

The pixels from odd tiles are written into the block of memory of a buffered tiller. When a row of an odd tile is finished, the pixels of the same row of an even tile are written into another block of memory of another buffered tiller. When the tile is half read, the JPEG 2000 core can be fed by the first 128 rows of pixels stored into the

memory of the buffered tiller. As a dual-port memory is used, the buffered tiller can be filled with the second half of the tile, in parallel with the JPEG 2000 core feeding operation. When a whole tile is processed (e.g. tiles 1 and 2 in Figure 6), a new tile can be processed by reusing the same buffered tiller (e.g. tiles 3 and 4 in Figure 6). The scheme shown in Figure 6 for processing the Y frame is replicated for U and V frames.

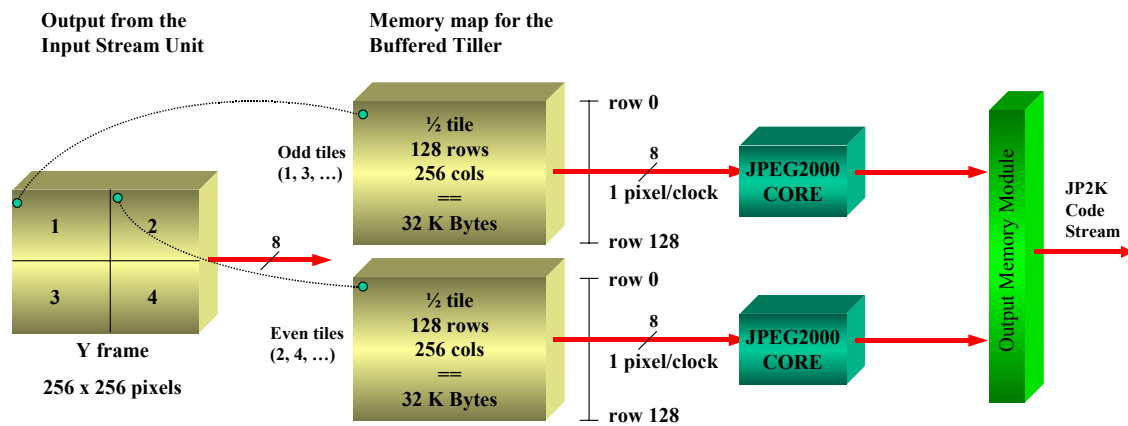


Figure 6. Tiling process.

A strategy for the frame sizes described in Table IX has not been designed yet. Several variables need to be analysed as, for instance, the use of an external memory module for buffering larger image sizes. A more detailed proposal for the design of the Tiller Unit will be released in a future revision of this report.

#### 4.5. Jpeg 2000 CODEC Unit

The Base Station provides the desired frame speed, and PUB calculates the best compression ratio to be used to deliver the image. The compression ratio is calculated as a function of the Memory Size used by the quantization table, the Frame Speed required by the Base Station, and the quantization table itself. There are five quantization tables available in the IP core, from lossless at 2x to lossy at 60x. According to the Jpeg 2000 CODEC Data Sheet, the best results are obtained by using a tile size of 256 x 256, precinct size of 128 x 128, and code-block size of 64 x 64. This corresponds to a memory of 140K. The idea is not to use the dynamic reconfigurability capabilities of the FPGA and, as a consequence, the Memory Size parameter in the following equations is to be fixed in 140K.

$$\text{Compression Ratio 1} = f(\text{Quantization Table 1, } \underline{\text{Frame Speed Required by Base Station}}, \text{Memory Size})$$

$$\text{Compression Ratio 2} = f(\text{Quantization Table 2, } \underline{\text{Frame Speed Required by Base Station}}, \text{Memory Size})$$

$$\text{Compression Ratio 3} = f(\text{Quantization Table 3, } \underline{\text{Frame Speed Required by Base Station}}, \text{Memory Size})$$

$$\text{Compression Ratio 4} = f(\text{Quantization Table 4, } \underline{\text{Frame Speed Required by Base Station}}, \text{Memory Size})$$

$$\text{Compression Ratio 5} = f(\text{Quantization Table 5, } \underline{\text{Frame Speed Required by Base Station}}, \text{Memory Size})$$

The Jpeg 2000 CODEC Unit programs the Jpeg 2000 IP core to use the quantization table that will provide the lowest compression ratio (better image quality), for the same frame speed. This strategy for compression ratio selection needs a deeper investigation.

#### **4.6. Output Stream Unit**

The design of the Output Unit, as well as the Tiller Unit, cannot be fully accomplished without further information about the JPEG 2000 IP core to be used. A vital piece of information related to the IP core is the input and output bandwidths. This piece of information is known for the IP cores investigated, when considering the use of only one core in the design. However, in order to increase the system performance, the core will be replicated to process the tiles of the input image in parallel. As the Tiller Unit has not been designed yet, at the moment it is not possible to define the number of cores to be used in parallel and, consequently, the output bandwidth of the JPEG 2000 CODEC Unit.

