Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

STV0974

Mobile Imaging DSP

Features

- **Supports VS6552 - 640 x 480 (VGA) color CMOS image sensor**
- **Supports VisionLink low EMI link to image sensor**
- **Specialized video processor for noise/defect filtering, color reconstruction, sharpness enhancement and radial corrections**
- **Programmable gamma correction for LCD support**
- **Programmable cropping, down-sizing by 1.5, 2, 2.5, 3, 4, 5 and 6, MMS (Multi Media Messaging Service) digital zoom**
- **JPEG compression, with programmable target file size**
- **M-JPEG operation at up to 30 frame/s at VGA resolution**
- **Programmable pixel output format including ITU-R 656 modes, RGB viewfinder modes and JPEG baseline**
- **Flashgun control**
- **Flexible host interface:**
  - 8-bit data /Hsync /Vsync video output interface and I²C camera control interface
  - 8-bit microprocessor interface with 2 Kbyte video FIFO for JPEG data, 10 Kbyte for non-JPEG data, interrupt and DMA requests
- **Multi-mode exposure control and color balance**
- **30 µW ultra low-power standby**
- **6 x 6 mm TFBGA low-footprint & lead-free package**

Description

The STV0974 is a low power digital image processor designed for the VS6552 color VGA image sensor. The STV0974 uses advanced image processing techniques to deliver high quality VGA images at up to 30 frames per second (frame/s). The sensor data received via the low EMI sensor interface is processed in real time: this includes pixel defect correction, color interpolation, image sharpness enhancement, selective noise filtering, cropping and scaling, allowing digital zoom for ViewFinder or MMS applications. Finally the image can be JPEG-compressed in real-time. The STV0974 also performs sensor housekeeping functions such as automatic exposure and white balance controls.

Applications

- **Mobile phone embedded camera system**
- **PDA embedded camera or accessory camera**
- **Wireless security camera**

Technical Specifications

Sensor	640 x 480 color CMOS (VS6552)
Frame rate (frame/s)	up to 30
Power supply	1.8 +/- 0.1 V
Power requirements	110 mW active < 30 µW standby
Package dimensions	6 mm x 6 mm x 1.2 mm
Temperature range	[-25; +70] °C

Ordering Information

Ordering code	Package
STV0974/TR	TFBGA SnPb balls
STV0974E/TR	TFBGA AFOP lead-free balls

Rev. 3

Contents

1 Overview

The STV0974 is a mobile imaging digital signal processor which, when used with VS6552 CMOS color VGA image sensor from STMicroelectronics, performs all the required data processing to deliver good quality Viewfinder, still and live color images. The STV0974 performs high quality color processing on images, achieving JPEG compression if requested and transfers them to a baseband through one of the available interfaces.

Data is transferred from sensor to STV0974 through Low Electromagnetic Interference (EMI) interface, using the sensor data transfer protocol over LVDS.

Data is transferred from STV0974 to Baseband

- through the video output interface. In video mode, the processor streams video data in a format which closely follows the data format specified in the ITU-R656 standard.
- through the microprocessor Interface. In microprocessor mode, the video data is stored in a small FIFO before is it pulled out of the asynchronous microprocessor interface by the host system (with DMA support).

1.1 Viewfinder mode

When connected to microprocessor interface or video output interface, the STV0974 can process Viewfinder image up to 30 frame/s.

1.2 Still features

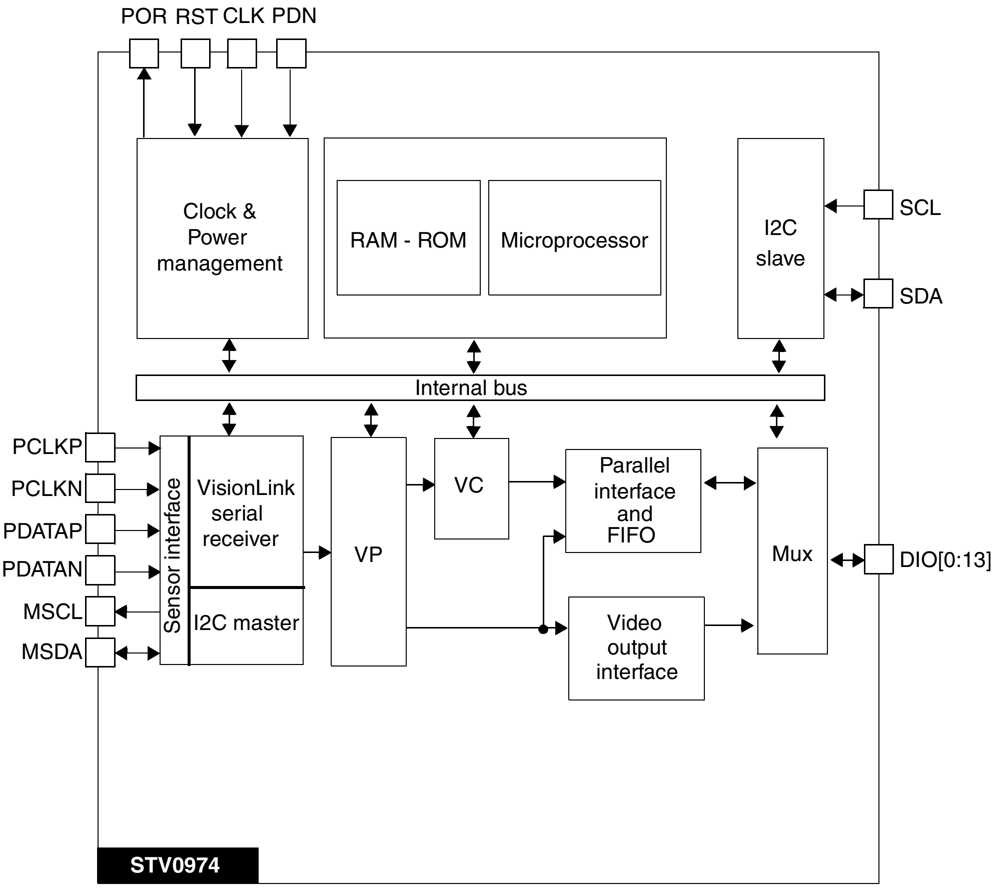
When requested by the baseband, the STV0974 captures bayer data from the sensor. Data is then color processed, down-scaled and/or compressed and sent through video output or microprocessor interface. In still mode, the first image produced has a guaranteed good exposure and color balance for single shot capture.

1.3 Live features

When connected to microprocessor interface or video output interface, the STV0974 can process live video up to 30 frame/s and eventually proceed to down-scaling and compression with on-chip Motion JPEG. Live mode is intended for capture of video sequences.

2 Functional block diagram

Figure 1: Functional block diagram



3 Signal description

Table 1: STV0974 signal description

Pin name	Type	Description
Power supplies		
VDD	PWR	Positive power supply
VCORE	PWR	Decoupling for internal core power supply
VDDPOR	PWR	Power on reset VDD supply 1.8 V
VSS	PWR	Digital ground
Sensor interface		
PDATAP, PDATAN	subLVDS In	Sensor data +, sensor data -, with internal 100 Ω termination resistor
PCLKP, PCLKN	subLVDS In	Sensor clock +, sensor clock -, with internal 100 Ω termination resistor
MSDA	I/O	Sensor I^2C data
MSCL	I/O	Sensor I^2C clock
Host interface		
POR	O	Power on reset output
RST	I	Reset input
CLK	I	System clock
PDN	I	Power down
DIO[0:11]DIO[13]	I/O	Host interface configurable I/O, see *Table 2*
DIO[12]	I/O	Flash Strobe Output (FSO)
SDA	I/O	Host I^2C data
SCL	I/O	Host I^2C clock
Test interface (ST internal use)		
TMS	I	Test mode
TCK	I	Test clock
TDI	I	Test data in
TDO	O	Test data out
Not connected		
NC		Not connected

Table 2: Host interface pins - output modes

Pin name	Microprocessor interface	Video port
DIO[0:7]	DATA[0:7]	DATA[0:7]
DIO[8]	RS	HSYNC
DIO[9]	CSN	VSYNC
DIO[10]	WRN	HCLK
DIO[11]	RDN	NC
DIO[12]	DRQ	FSO[a]
DIO[13]	IRQ	NC

[a] Flash Strobe Output

4 Functional description

4.1 Overview

The processor includes a chain of dedicated video data processing blocks controlled by a microprocessor. The processing blocks perform the main video pipe processing while the microprocessor manages the interactions between the sensor, the functional blocks and the host.

The host controls and monitors the STV0974 via a set of read/write registers accessible via the I^2C interface for the streaming video mode and via the asychronous microprocessor interface for the microprocessor mode.

In video mode, the processor streams video data in a format which closely follows the data format specified in the ITU-R656 standard.

In microprocessor mode, the video data is stored in a small FIFO before is it pulled out of the asynchronous microprocessor interface by the host system (with DMA support).

4.1.1 Video pipe block description

Please refer to the block diagram (*Figure 1*).

Sensor interface This block decodes the incoming serial data stream from the sensor (raw bayer data) and converts it into a parallel form for the processing chain.

Video processor The video processor converts the raw bayer data from the sensor to RGB or YUV processed data by applying a number of filters to the data then scaling and converting the data into either one of the RGB modes or into YUV mode.

Video JPEG compressor The video compressor converts the processed data from the video processor and converts the data into JPEG format. The compression ratio applied to the image can be controlled by the microprocessor.

Streaming video output port In streaming video mode the data from either the video processor or the video compressor is enclosed in a format which closely follows the data format specified in the ITU-R656 standard.

Microprocessor interface In microprocessor interface mode, the data from the video processor or video compressor is stored in a FIFO. The interface informs the system via an IRQ or a DRQ that the FIFO is filling up. The system then has to pull some of the data from the STV0974 via the microprocessor interface.

4.1.2 Control

Register map The STV0974 is controlled via a register map that is maintained by the STV0974 microprocessor. Each register in the map has an address and contains either read or read/write data. The read only registers detail the current state of the STV0974. Read/write registers can be written to in order to modify the default behavior of the STV0974. The map is accessed via I^2C or via the microprocessor interface.

Micro processor interface In microprocessor interface mode the STV0974 register map can be accessed by writing the address of the register to the port and then reading or writing the register value.

Video output interface In streaming video mode, the STV0974 register map can be accessed via the I^2C port on the STV0974. The STV0974 is addressed by supplying the device address, register address and value to be written or read.

Microprocessor The microprocessor maintains the system interface via the register map. Any changes in system state are reflected in this map by the microprocessor and any changes commanded by the host system via this interface are then applied by the microprocessor.

When the system is commanded to change state, the microprocessor configures the functional blocks from the STV0974 and the sensor into the requested mode. The register map is updated accordingly to reflect the new state of the hardware.

The microprocessor monitors statistics gathered from the incoming image data and responds to changes in images. It adapts the functional block settings to correct for shifts in environmental conditions such as light level and illumination color temperature. The microcontroller will optimize these settings to provide the best quality image on all occasions.

4.1.3 Other functional blocks

Power management The hardware state of the STV0974 can be controlled by the power down pin (PDN). Upon the application of power to the STV0974 and PDN release, the STV0974 power-on-reset cell issues a timed reset pulse and then releases the STV0974 into its boot state.
The power-on-reset cell which output is the POR signal, is externally connected to the RST pin.

Clocks In sleep mode, the STV0974 clock is derived from the clock signal applied to the CLK pin. In all other modes, the STV0974 clock is derived from the high speed clock received from the sensor.

4.2 Sensor interface

4.2.1 Features

- Low electromagnetic interference (EMI) interface with CMOS image sensors
- High speed serial receiver, with data and clock inputs
- Up to 120 Mbit/s operation using very low voltage differential signaling (vLVDS)
- VisionLink transfer protocol
- I^{2}C compliant master controller, 1.8 V interface, up to 400 kHz operation

4.2.2 Description

The STV0974 sensor interface is dedicated to the VS6552 image sensor that uses the VisionLink data transfer protocol over vLVDS. This includes:

- An I^{2}C master controller supporting 1.8 V interface and 400 kHz operation. The I^{2}C master port signals are MSDA and MSCL that require external pull-up resistors. Internally, the I^{2}C master is a peripheral of the microprocessor control unit.
- Two vLVDS receivers for sensor data and clock signals, PDATA and PCLK differential pairs respectively. Each receiver accepts 1.8 V LVDS signals
- A VisionLink data synchronization and extraction unit, which extracts image timing references, active video and sensor status information. The extracted video stream in raw Bayer format along with active video strobes are connected to the video processing unit. The sensor status information is presented to the microprocessor control unit.

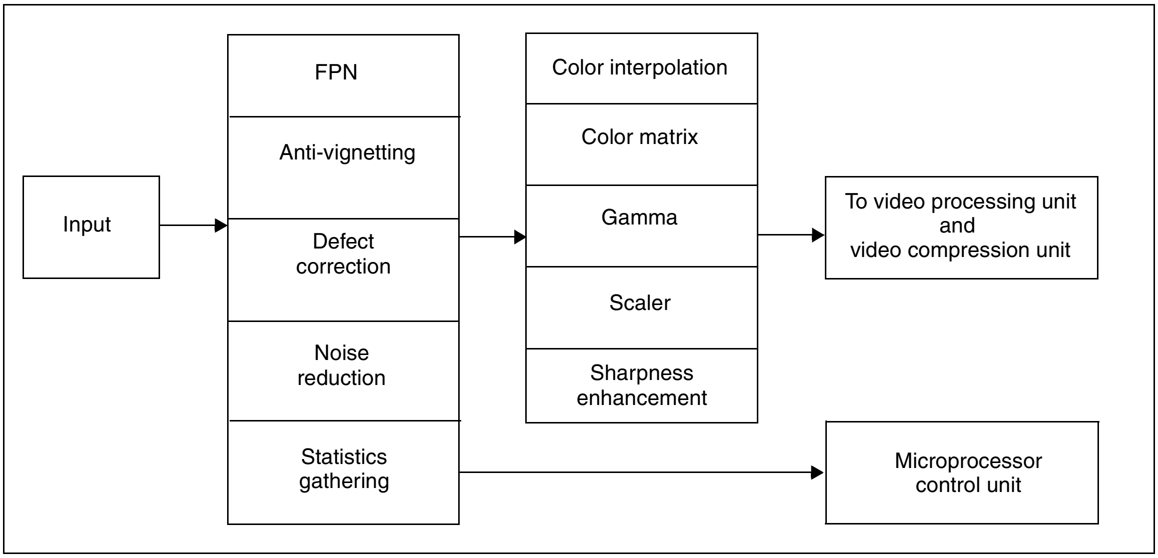
4.3 Video processing unit

4.3.1 Features

- Low-power dedicated hardware video processing unit, pipeline operation up to VGA resolution 30 Hz
- Image sensor correction stage including pixel defect correction and fixed pattern noise (FPN) cancellation
- Color interpolation stage with anti-aliasing and color matrix compensation
- Optical system compensation stage including anti vignetting and sharpness enhancement
- Noise reduction filter
- Programmable gamma and s-curve gamma for LCD support
- Full frame statistics gathering for exposure and color balance controls
- Programmable output image size (downscale by 1.5, 2, 2.5, 3, 4, 5 and 6)

4.3.2 Overview

Figure 2: Video processing unit



Fixed Pattern Noise (FPN) cancellation The FPN cancellation algorithm removes any column variability over the video area.

Statistics gathering Image statistics are gathered on the full resolution input image and forwarded to the camera control unit for exposure and color balance control loops.

Anti vignetting A radial gain is applied to the image luminance to compensate for possible luminance loss in the corners of the image due to an imperfect lens system.

Defect correction The defect correction algorithm can detect and correct any defective pixels in a sensor array.

Noise reduction filter The noise reduction filter is based on an adaptive algorithm. This algorithm performs filtering but does not affect image areas including significant information.

Color interpolation Each pixel RGB components are calculated by interpolation of the incoming Bayer pattern.

Color matrix Each pixel (RGB vector) is multiplied by a color matrix to adjust color balance. Viewfinder and live settings are independent to allow for optimization of both LCD display and capture for later viewing (i.e. on a PC).

Sharpness enhancement A sharpening two-dimensional mask is applied to Green only and from interpolation. The resulting data is added (with a gain factor) to the matrix RGB data.

Downscaler The downscaler unit extracts a rectangular region of interest and resizes the image by resampling video data. Standard image size such as CIF, QVGA and QCIF are available as well as a fully programmable custom size:

Table 3: Standard image size, VGA input

Format	Image size	Cropping	Scaling	Comments
VGA	640 x 480	None	None	
CIF	352 x 288	528 x 432	/ 1.5	82.5% of input image used, centered
QVGA	320 x 240	None	/ 2	
QCIF	176 x 144	528 x 432	/ 3	82.5% of input image used, centered
QQVGA	160 x 120	None	/ 4	
SubQCIF	128 x 96	None	/ 5	
QQCIF	88 x 72	528 x 432	/ 6	82.5% of input image used, centered
Custom	max VGA	Any	Any	See below

When custom size is selected, the crop and scale parameters are subject to the following constraints to ensure proper operation:

- Output image size must be in 8 x 8 pixels increments
- Scaling factor can be any value giving an input image size within input limits

Gamma correction A non-linear gain is applied to each pixel's RGB components to compensate for the display's non-linearity. A standard curve is available for image capture for later viewing on a PC and an S-curve is available for LCD display.

Coder The coder unit converts the internal RGB video stream to a user selectable output video format. It is based on a YUV digital video encoder with embedded synchronization codes, compliant with *[1]*, extended with the support of RGB formats for viewfinder usage, as shown in *Table 4*.

Table 4: Output video formats

Name	Format	Description
UYVY	$Y_7Y_6Y_5Y_4Y_3Y_2Y_1Y_0$ $U_7U_6U_5U_4U_3U_2U_1U_0$ $Y_7Y_6Y_5Y_4Y_3Y_2Y_1Y_0$ $V_7V_6V_5V_4V_3V_2V_1V_0$	YUV (or YC_BC_R) 4:2:2 format as per [1]
RGB565	$R_4R_3R_2R_1R_0G_5G_4G_3$ $G_2G_1G_0B_4B_3B_2B_1B_0$	16-bit RGB format for direct viewfinder on 64 K color LCDs.
RGB444	$0_30_20_10_0R_3R_2R_1R_0$ $G_3G_2G_1G_0B_3B_2B_1B_0$	16-bit RGB format for direct viewfinder on 4096 color LCDs. Uses 4 bit per RGB component, the four MSB's are zero padded.
RGB332	$R_2R_1R_0G_2G_1G_0B_1B_0$	8-bit RGB format for low bit rate viewfinder usage.

Byte ordering assumes a little endian memory system, i.e. in 16-bit formats, the least significant byte is sent first. For example, the UYVY format produces the sequence U Y_0 V Y_1... as per [1].

Note: *Nevertheless various options are available to suit memory system requirements:*

- Byte ordering can be changed to big endian
- In YUV formats, the U and V components can be swapped
- In RGB formats, the R and B components can be swapped

YUV format processing The RGB pixel is converted to YUV coordinates according to ITU-R BT601 specification. The YUV coordinates are then rounded and clipped for an 8-bit representation. To produce a 4:2:2 digital component video, U and V components are filtered and down sampled by a factor of 2, coincident with Y sampling time.

RGB format processing Dithering: In order to avoid contouring effects on low color depth displays, the RGB components are dithered prior to truncation to the required number of bits.

Framing: The output frame is produced by performing the following steps:

1 **Blanking code insertion:** During video blanking intervals, blanking codes are inserted in the output stream. The default blanking code is the 16-bit pattern 0x1080, corresponding to Y = 0x10 and U/V = 0x80 as per [1].

2 **Synchronization pattern detection and correction:** The coder performs detection of various synchronization patterns and applies a correction according to the current output format.

3 **Video Timing Reference Code Insertion:** A 4-byte sequence is inserted at the beginning and the end of each digital video line to delineate lines and frames in the video stream. The sequence is defined in [1] as FF 00 00 XY, where the XY byte is defined by:

Table 5: XY bits definition

Bit	Symbol	Definition
7 (msb)	1	Always 1.
6	F	Even / Odd Field. To maintain compatibility with[1], F is alternatively 0 or 1.
5	V	V = 1 during field blanking, 0 otherwise.
4	H	H = 1 during line blanking, 0 otherwise.
3	P3	Protection bit: P3 = V xor H
2	P2	Protection bit: P2 = F xor H
1	P1	Protection bit: P1 = V xor V
0 (lsb)	P0	Protection bit: P0 = F xor V xor H

SAV (Start of Active Video) is defined as the 4-byte sequence where H = 0.

EAV (End of Active Video) is defined as the 4-byte sequence where H = 1.

4.4 Video compression (VC)

Real time video compression permits a frame rate of 30 frame/s in any mode at VGA.

The JPEG compression engine is a standard baseline sequential JPEG encoder [2].

The compression ratio can be modified by applying a multiplication factor on the quantization table. The quantization table can be scaled from a factor of 1/8 to a factor of 8.

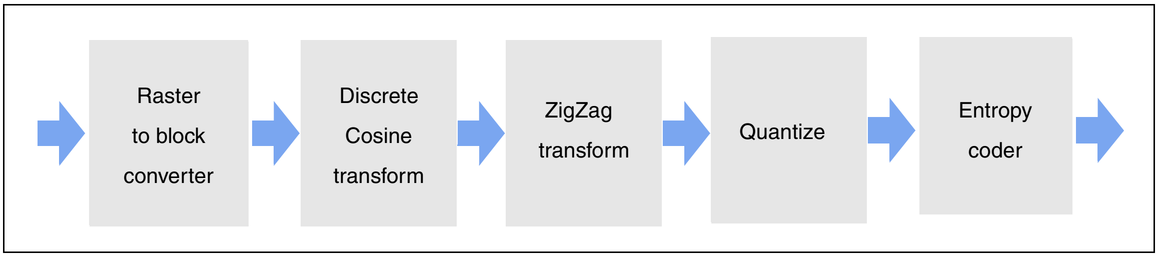
The STV0974 video compression block includes a baseline DCT JPEG encoder compliant with ISO DIS 10918-1.

The JPEG encoder has the following characteristics:

- baseline sequential DCT based encoder
- YUV 422 encoding only
- up to VGA image size
- scalable quantization table
- standard quantization table
- standard Huffman coder

The encoder top level block diagram is presented in *Figure 3*.

Figure 3: Encoder top level block diagram

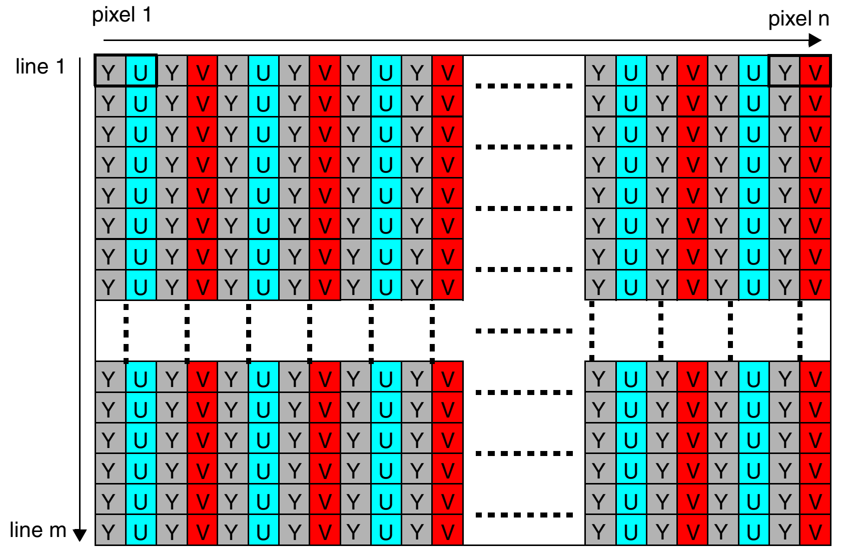


The input data is a YUV 422 8-bit data stream in raster order. The output data is a baseline JPEG data stream.

4.4.1 Raster to block converter

This block transforms the raster scan ordered data into block based ordered data. This data ordering is compliant with ISO DIS 10918-1 Annex A - Section A.2.

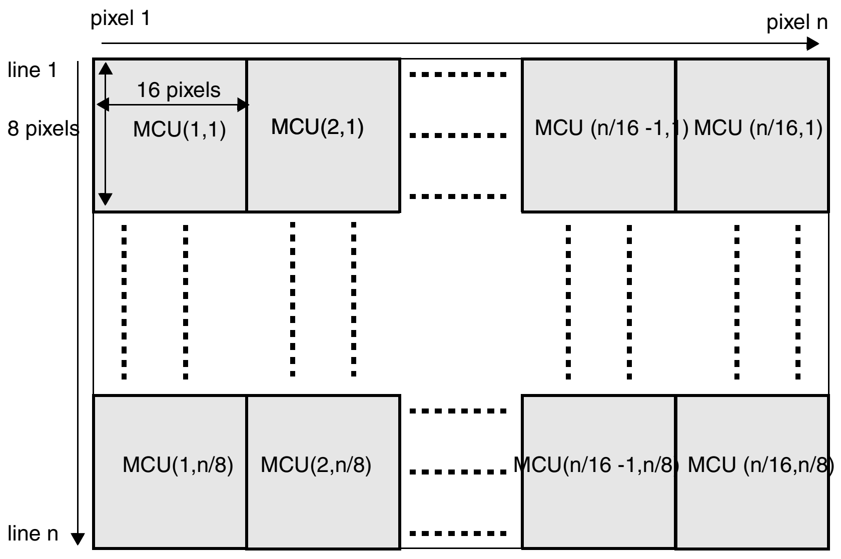
Figure 4: Data sequence at Raster to block input



The sequence of the input data stream is the following: line 1 from left to right up to pixel n, then line 2 from left to right......up to line m, pixel n.

The output data stream sequence is block based. The image is segmented into MCU (minimum coded units) as illustrated in *Figure 5*.

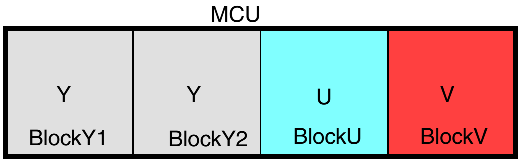
Figure 5: MCU data order



The MCU sequence order is top left to top right and top to bottom.

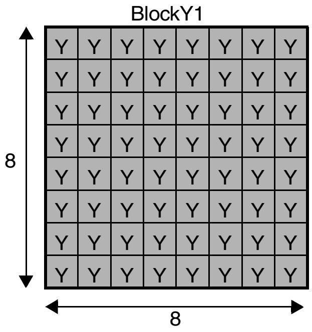
Figure 6 shows the MCU structure made of 4 blocks: 2 blocks of 8x8 Y component, 1 block of 8x8 U component and one block of 8x8 V component. The series of blocks must be processed according to this order.

Figure 6: Structure of each MCU



Each block is composed of 8x8 components. *Figure 7* presents the structure of BlockY1, as an example.

Figure 7: Structure of block Y1



Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	Y	Y	Y	Y	Y	Y

The data sequence inside each block is left to right and top to bottom.

To summarize, at the output of Raster to Block converter, the data order is the following:

Y data of blockY1 of first MCU (64 data from left to right, then top to bottom)

Y data of blockY2 of first MCU (64 data from left to right, then top to bottom)

U data of blockU of first MCU (64 data from left to right, then top to bottom)

V data of blockV of first MCU (64 data from left to right, then top to bottom)

Y data of blockY1 of second MCU (64 data from left to right, then top to bottom)

Y data of blockY2 of second MCU (64 data from left to right, then top to bottom)

U data of blockU of second MCU (64 data from left to right, then top to bottom)

V data of blockV of second MCU (64 data from left to right, then top to bottom)

... up to last image MCU.

4.4.2 Discrete Cosine Transform

This block performs a Discrete Cosine Transform on the incoming data stream. It is compliant with ISO DIS 10918-1 Annex A - Section A.3.

The block processes each 8x8 input block to transform them into 8x8 DCT coefficients. The calculation of the DCT coefficients is done by the formula:

$$F(u, v) = \frac{2}{N} \times \sum_{x=0}^{7} \sum_{y=0}^{7} C(u)C(v)f(x, y)\cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}$$

with

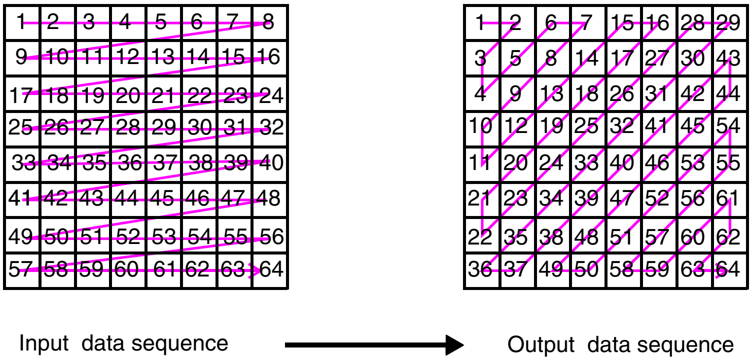
$$C(u), C(v) = \frac{1}{\sqrt{2}} \qquad \forall(u, v) = 0$$

$$C(u), C(v) = 1 \qquad \forall(u, v) \text{ ¼ } 0$$

4.4.3 Zigzag transform

This block is in charge of setting the DCT coefficients in a sequence that corresponds to an increasing spacial frequency of the cosine function. It is compliant with ISO DIS 10918-1 Annex A - Section A.3.6.

Figure 8: ZigZag block sequence re-ordering



1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

1	2	6	7	15	16	28	29
3	5	8	14	17	27	30	43
4	9	13	18	26	31	42	44
10	12	19	25	32	41	45	54
11	20	24	33	40	46	53	55
21	23	34	39	47	52	56	61
22	35	38	48	51	57	60	62
36	37	49	50	58	59	63	64

4.4.4 Quantization block

This block applies a uniform quantizer on all DCT coefficients, in ZigZag sequence. It is compliant with ISO DIS 10918-1 Annex A - Section A.3.4.

The quantizer step size for each DCT coefficient S_{uv} is the value of the corresponding element Q'_{uv} from the quantization table Q'.

$$Squv = round\left(\frac{Suv}{Q'uv}\right)$$

Where *uv* is the index of the zigzag coefficient.

Table Q' is a scaled quantization table calculated for table Q as follows:

$$Q' = \frac{Squeeze}{32} \times Q$$

where *Squeeze* is a parameter value.

Table Q is represented in *Figure 9*, as described in ISO DIS 10918-1 Annex K.

Figure 9: Luminance and chrominance quantization tables

$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$	$Q = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}$
Quantization table for Y blocks	Quantization table for U and V blocks

Table 6 shows an example of VGA image when different squeeze values are applied by the user.

Table 6: VGA image size - YUV 4: 2: 2 - Example of image size after JPEG compression

Squeeze	[a]	2	4	8	10	20	30	50	67
Squeeze quantization table factor (Squeeze/8)		0.12	0.25	0.5	0.62	1.25	1.87	3.12	4.19
File size (Kbyte)	614	~80	~51	~32	~27	~21	~12	~9	~9
Bit per pixel		2.13	1.39	0.85	0.72	0.56	0.32	0.24	0.24

[a] No compression

4.4.5 Entropy coder

This block performs the following functions:

- insertion of JPEG Markers
- runlength encoding
- Huffman encoding

4.4.5.1 JPEG markers

These markers are compliant with ISO DIS 10918-1 Annex B.

The output JPEG file includes markers defined in *Table 7*, in order of appearance.

Table 7: JPEG markers included in STV0974 **output data stream**

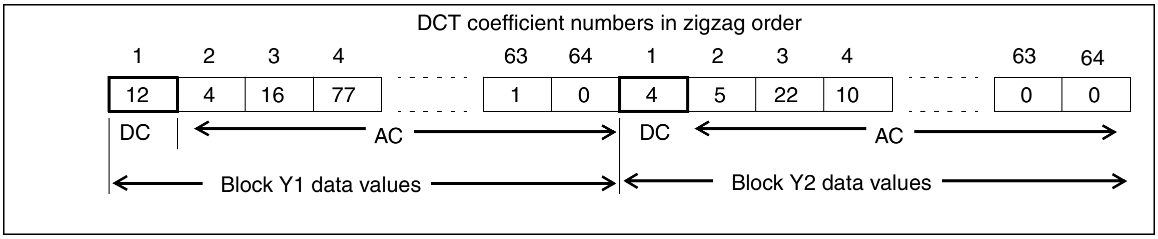
Marker Function	Name	Value
Start of image	SOI	FFD8
Define Quantization Table	DQT	FFDB
Start of frame for baseline DCT	SOF_0	FFC0
Define Huffman Tables	DHT	FFC4
Start of Scan	SOS	FFDA
End of image	End of image	FFD9

4.4.5.2 Runlength and Huffman encoding

Encoding of DC coefficient

The so-called DC coefficient is the first coefficient of each DCT data block. This DC coefficient is coded through its DPCM difference with its previous value, which is huffman encoded. This is described in ISO DIS 10918-1 Annex A - Section F.1.2.1. The DC Huffman tables are described in ISO DIS 10918-1 Annex A - Section K.3.

Figure 10: Encoding of DC coefficient



In the example from *Figure 10*, the DC coefficient in Block Y2 is equal to 4, the previous Luminance DC coefficient is 12 (DC coefficient of Block Y1). The DPCM value is 4-12 = -8 and the encoded value will be Huffman (-8). The code that is generated is Code = DC Huffman (-8).

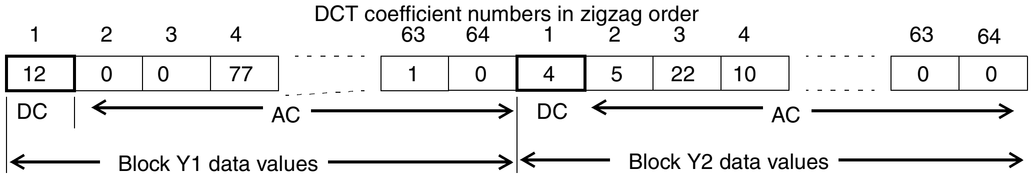
Encoding of AC coefficients

The 63 left coefficients of each DCT block are called AC coefficients. They are encoded using run-length and Huffman encoder. The run-length encoding consists in counting the number of zero values between each non-zero coefficient. When a non zero coefficient is found, the Huffman code of the pair (number of preceding zero, Number value) is Huffman encoded. If a run contains more than 15 zeros, a specific number called ZRL is Huffman encoded.

If all the values up to the end of the block are equal to zero, a specific code called EOB is Huffman encoded.

The Huffman table used are described in ISO DIS 10918-1 Annex A - Section K.3.

Figure 11: Encoding of AC coefficient



In the above example, the first AC coefficient of Block Y1 is 0, as good as the second one. The zeros are not Huffman encoded, but the runlength counts them. When the first non-zero value is reached (Coefficient 4 with value 77), the Huffman code for the pair (number of preceding zeros, value) = (2,77) if Huffman encoded.

The code that is generated is Code = Huffman (2,77).

4.5 Microprocessor interface

4.5.1 Features

- 8-bit microprocessor interface, asynchronous read/write, one address bit
- Indirect access to image sensor and coprocessor control registers
- Direct access to image data (JPEG compressed or uncompressed)
- On-chip 2048 byte image FIFO
- Interrupt request output
- 8/16/32-byte burst DMA support
- 2 Kbyte video FIFO for JPEG data and 10 Kbyte FIFO for non-JPEG data

4.5.2 Description

The STV0974 can be connected to any general purpose 8-bit microprocessor via the microprocessor interface. This interface substitutes functionally to the YUV and I²C interfaces, i.e. both data and control flows are handled through the interface which provides:

- access to the image data FIFO for fast transfers of scaled-down viewfinder images or full-resolution captured and compressed image data. For host systems with DMA support, a DMA request line is provided, as well as programmable FIFO threshold for burst operation. For other systems, an interrupt request output line is provided.The 2048-byte FIFO allows for greater host system latencies; to suit system requirements, the FIFO threshold is programmable.
- access to the camera subsystem configuration and control registers, through an address/data register pair and a status register for data polling. Access requests are posted to the internal controller core that handles the request (as in I²C mode) and finally acknowledges through the microprocessor interface status register.

4.5.3 Direct registers

Access to the microprocessor interface direct registers is controlled by the state of CSN, RDN, WRN and RS (*Table 8*).

Table 8: Microprocessor Interface Direct Registers

CSN	RDN	WRN	RS	Register accessed
0	1	0	0	Address Register (AR)
0	0	1	0	Status Register (SR)
0	1	0	1	Data Write register (DW)
0	0	1	1	Data Read register (DR)
1	X	X	X	No access

The direct registers are used to access all STV0974 indirect registers and external image sensor registers through I²C. To read from a camera register:

1 Write AR with the indirect register address.

2 Poll the status register RDY bit until high.

3 Read the register data from DR.

To write to a camera register:

1 Write AR with the indirect register address.

2 Write DW with the register data.

3 Poll the status register RDY bit until high.

Note: 1 16-bit values are in little-endian representation, i.e. LSB at lower address.

2 No data polling is required to access the microprocessor interface indirect registers.

Address Register (AR)

The Address Register holds the 16-bit address of the camera register to access. AR is written by two consecutive byte writes, least significant byte first.

Note: To avoid LSB/MSB sequence mismatch, any read access (to DR or SR) guarantees that the following write to AR updates the LSB (ADDR bits 7:0).

Table 9: Address Register

Bits	Name	Type	Description
15	ME	WO	0 = Image sensor register (forward command to I²C master). 1 = STV0974 register.
14	RW	WO	0 = Write access 1 = Read access
[13:0]	ADDR	WO	Camera register address.

Status Register (SR)

The status register is an 8-bit read-only direct register holding all pending requests from the camera subsystem.

Table 10: Status Register

Bits	Name	Type	Description
7	IRQ	RO	Interrupt Request: IRQ is set when at least one of the interrupt sources is set, and the corresponding bit mask is set.
6	-	-	Reserved.
5	EOF	-	End Of Frame: EOF is set by the falling edge of VENV (output image vertical envelope). SOF is cleared by writing ICLR bit 5.
4	SOF	RO	Start Of Frame: SOF is set by the rising edge of VENV (output image vertical envelope). SOF is cleared by writing ICLR bit 4.
3	MCI	RO	Micro-Core Interrupt: MCI is set by the micro-core to alert the host of the occurrence of an internal event (status update, error, etc.…). MCI is cleared by writing ICLR bit 3.
2	FERR	RO	FIFO Error: FERR is set by the FIFO controller if a FIFO overflow occurs, or if the FIFO is not empty when cleared at the start of frame. FERR is cleared by writing ICLR bit 2.
1	FRDY	RO	FIFO Ready: This bit indicates that the number of valid bytes in the FIFO is greater than or equal to the FIFO threshold value, i.e: FRDY = (Nbytes ≥ threshold) During the inter-frame period, ‘threshold’ is forced to ‘1’ to flush the FIFO; otherwise, ‘threshold’ is determined by FHTR. FRDY is level sensitive, i.e. it can be cleared only by reading FIFO.
0	RDY	RO	Ready: This bit indicates the state of the access request between the host and the STV0974: 0 = Register access in progress, 1 = AR, DR and DW can be accessed by the host. For a read access, RDY is cleared upon host write to AR (MSB); it is set by the micro-core when DR is updated with the register data. For a write access, RDY is cleared upon host write to DW; it is set by the micro-core when the internal register is updated. Note: RDY is high when AR points to the interface indirect registers.

Data Write Register (DW)

The data write register contains the byte to transfer to a camera register. DW can be written only when SR bit RDY is set.

Table 11: Data write register

Bits	Name	Type	Description
[7:0]	DW	WO	Data Byte to write to camera subsystem.

Data Read Register (DR)

The Data Read Register contains the byte transferred from a camera register. DR is valid only when SR bit RDY is set.

Table 12: Data Read register

Bits	Name	Type	Description
[7:0]	DR	RO	Data Byte read from the camera subsystem.

4.5.4 Indirect registers

The microprocessor interface indirect registers are accessed by the host using an indirect address base of 0x8FF0 / 0xCFF0 (write / read). Register offsets are listed in *Table 13*:

Table 13: Microprocessor interface indirect register map [a] [b]

Offset	Name	Description
0x00	FIFO	FIFO read register.
0x01	MICR	Microprocessor interface control register
0x02	IMASK	Interrupt mask register
0x03	ICLR	Interrupt clear register
0x04 0x05	FTHR	FIFO threshold register.
0x06 0x07	FCNT	FIFO count register.

[a] 16-bit values are in little-endian representation, i.e. LSB at lower address.

[b] No data polling is required to access the microprocessor interface indirect registers.

FIFO Register (FIFO)

FIFO is a read-only register. When read, FIFO returns the least recent byte from the image data FIFO, decrements the byte count and releases the FIFO interrupt if the count is lower than the threshold. Reading from an empty FIFO returns the last valid byte read.

The image data FIFO is cleared at the beginning of VENV, the image vertical envelope. If the FIFO is not empty, its contents are discarded and the FERR flag is raised in the status register SR. New image data start to fill in the FIFO. If an overflow occurs during VENV, the FERR flag is also raised in SR; FERR can be cleared through ICLR.

Table 14: FIFO register

Bits	Name	Type	Description
[7:0]	FIFO	RO	Image data byte (uncompressed or compressed).

Microprocessor Interface Control Register (MICR)

MICR controls and configures the image data transfer.

Table 15: Microprocessor Interface Control Register

Bits	Name	Type	Description
[7:6]	-	-	Reserved.
5	IRQPOL	RW	IRQ pin polarity: 0 = active high 1 = active low
4	DRQPOL	RW	DRQ pin polarity: 0 = active high 1 = active low
[3:2]	BSIZE	RW	DMA burst size and enable: 00 = DMA operation disabled, DRQ pin is high impedance 01 = 8-byte burst 10 = 16-byte burst 11 = 32-byte burst
1	-	-	Reserved, read as zero, ignored upon write
0	CLR	WO	Clear FIFO (Write Only, read as 0): 0 = No action 1 = Reset FIFO to empty state

Interrupt Mask Register (IMASK)

Table 16: Interrupt Mask Register

Bits	Name	Type	Description
[7:6]	-	-	Reserved
[5:0]	IMASK	RW	Each IMASK bit set to ‘1’ enables the corresponding interrupt source bit in the status register (SR)

Interrupt Clear Register (ICLR)

Table 17: Interrupt Clear Register

Bits	Name	Type	Description
[7:6]	-	-	Reserved
[5:2]	ICLR	WO	Each ICLR bit written with a ‘1’ clears the corresponding interrupt source bit in the status register (SR). Writing a ‘0’ has no effect
[1:0]	-	-	Reserved

FIFO Threshold Register (FTHR)

Table 18: FIFO threshold register

	Description	
FTRH	Holds the FIFO threshold value	
NE = 1	threshold = 1	(TH is ignored)
NE = 0	threshold = TH * 16	(TH valid range is [1, 2...127])

This register is used to program values such as 1 (flush), 16 or 32 (DMA burst) or any greater value up to 2032 for interrupt driven data transfer. Note that for proper DMA operation, ‘threshold’ must be greater than or equal to the DMA burst size (MICR[BSIZE]).

Table 19: FIFO Threshold Register

Bits	Name	Type	Description
[15:11]	-	-	Reserved
[10:4]	TH	RW	Threshold value in 16-byte increments.
[3:1]	-	-	Reserved
0	NE	RW	Not Empty: 1 = Force threshold to 1 (TH is ignored) 0 = Normal