JPEG Decoder

Description

The 29C82 performs decompression of still pictures in accordance with the ISO JPEG 8 R6 standard for basic systems (pixels coded in 8 bits, sequential display, HUFFMAN coding, etc). The 29C82 independently executes most following of the adaptive discrete cosine transform (ADCT) algorithm :

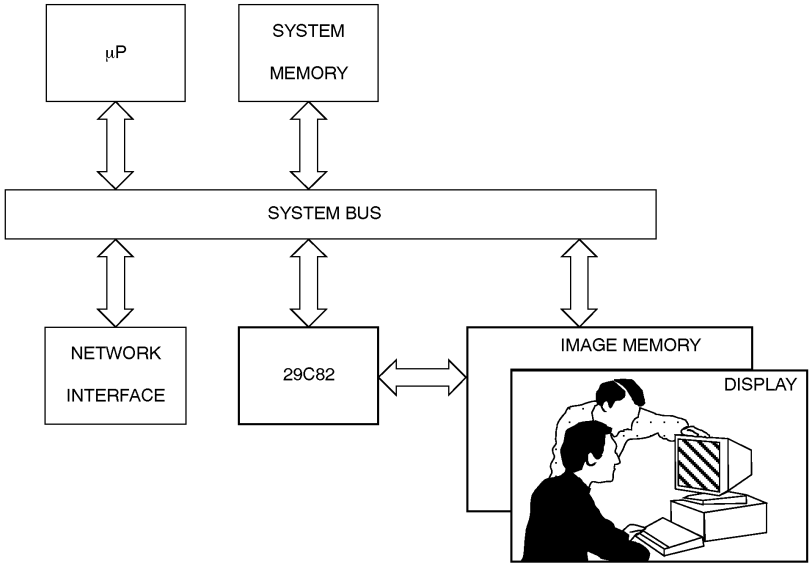
– VLC (HUFFMAN) decoding using tables created by the user or contained in the JPEG image frame. Two AC tables and two DC tables can be loaded into the 29C82.
– de-quantization using tables contained in the JPEG image frame. Two DC tables and two AC tables can be loaded into the 29C82.
– inverse COSINE transform.

The 29C82 is a peripheral circuit which communicates with the microprocessor over the DATA [0..7] and ADDRESS [0..3] busses. The microprocessor performs decoding of the JPEG image frame header in order to extract the markers, the image parameters, the quantization, and the VLC tables. These tables and parameters are then used to program the 29C82. After decoding, the pixels are available on pixels bus PIXEL[0..7] of the 29C82 or on the DATA [0..7] system bus. The 29C82 is able to provide pixels at a speed of 1.25 MPixels/s on PIXEL[0..7], with the system clock (CLK) at 20 MHz.

Features

- Decoding of still images to the ISO JPEG-8 Rev6 standard.
- Performs VLC decoding, de-quantization and inverse discrete cosine transform (IDCT).
- Maximum system clock frequency : CLK = 20 MHz.
- Maximum output bit rate : 1.25 mega-pixels/s on the private bus.
- 8-bit microprocessor interface.
- Dedicated 8-bit pixel bus
- 5 V DC power supply.
- Package : PLCC68

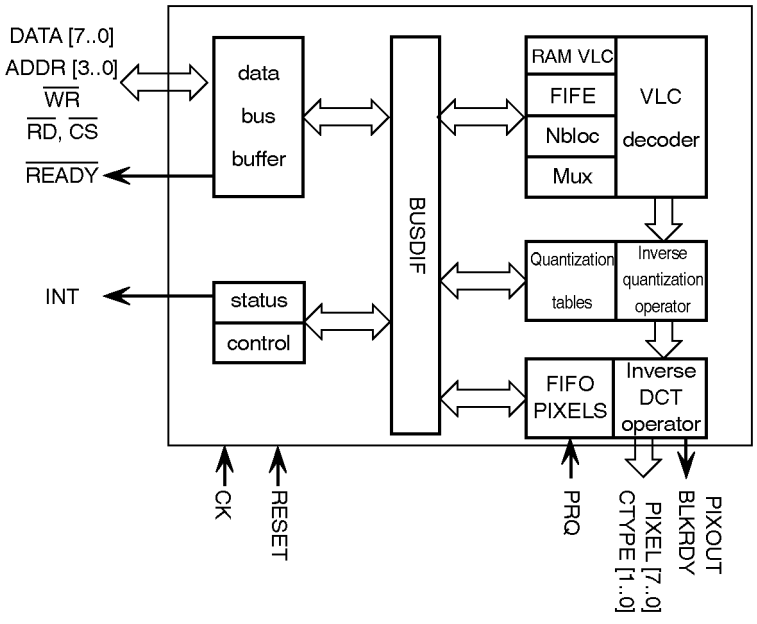
Figure 1. Application Block Diagram using 29C82



* CCETT License
This data sheet includes information obtained from CCETT

Architecture

Figure 2. 2982 Block Diagram



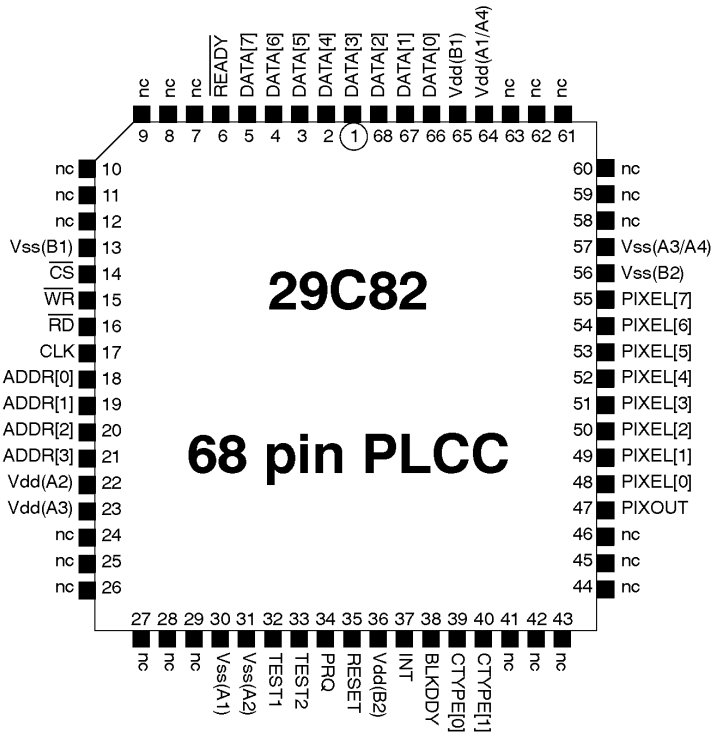
The 29C82 is a peripheral circuit, the system bus of which is compatible with a demultiplexed address/data bus of the INTEL type. It contains 4 blocks which are interconnected over the internal bus (BUSDIF). The internal “VLC decoder”, “de-quantization operator” and “DCT operator” blocks are also interconnected over a data path which enables the passage of coefficients between the various blocks to be optimized, and the traffic on the internal BUSDIF bus to be reduced. After decompression over the various stages mentioned previously, the pixels can then be read by the system bus (over BUSDIF) or on the PIXEL[7..0] private bus.

The 29C82 includes the following blocks :

– the VLC block :
  - VLC decoder (see appendix 3).
  - 2 VLC DC tables, each containing 32×5 bits.
  - 2 VLC AC tables, each containing 512×9 bits.
  - 2 sets of registers (Nblock and Mux)
– the de-quantization block
  - the de-quantization operator (see appendix 1).
  - 4 de-quantization tables, each containing 64 levels.
  - 1 programming register (PROGRAMQ)
– the IDCT block :
  - 1 inverse DCT operator of the Duhamel-Guillemot type
  - 1 output FIFO with two 64-byte pages working alternately - each page can be accessed by the DATA[7..0] system bus or by the PIXEL[7..0] bus (like a single 64-bit FIFO).
– the microprocessor interface
  - 1 interrupt monitor with status and control registers.
  - 1 interface block to allow connection to a microprocessor over the DATA[7..0] bus, the ADDR[3..0] bus, WR, RD, CS and READY.

Pin Configuration

Figure 3. 29C82 68 pin PLCC Package. Top View



NAME	TYPE	FUNCTION
Vdd (A1/A4) Vdd (A3) Vdd (A2) Vdd (B2) Vdd (B1)	supply supply supply supply supply	core positive supply (+ 5V) core positive supply (+ 5V) core positive supply (+ 5V) buffer positive supply (+ 5V) buffer positive supply (+ 5V)
Vss (A1) Vss (A2) Vss (A3/A4) Vss (B1) Vss (B2)	supply supply supply supply supply	core negative supply (0V) core negative supply (0V) core negative supply (0V) buffer negative supply (0V) buffer negative supply (0V)
CLK	I	reference clock
RESET	I	hardware reset (active high) table contents are not changed by reset
DATA[7..0]	I/O	8 bit system DATA bus
ADDR[3..0]	I	4 bit system ADDRESS bus
$\overline{CS}$	I	chip select active low
$\overline{RD}$	I	READ active low
$\overline{WR}$	I	WRITE active low
$\overline{READY}$	O	READY output signal active low

NAME	TYPE	FUNCTION
INT	O	interrupt signal active high
PRQ	I	pixel request used to request from 29C82 a block of pixels on pixel bus (active high)
PIXEL[0..7]	O	8 bit tristate pixel bus (at CLK frequency)
CTYPE[1..0]	O	component type (tree state) user defined code (ex : cyan, green, Y)
BLKRDY	O	block ready, indicates that a new 64 pixel block is available (active high)
PIXOUT	O	pixel out flag, indicates that pixels are present on pixel bus (active high)
TEST1	I	TEST pin, should be left floating in normal mode
TEST2	I	TEST pin, should be left floating in normal mode

Description of Inputs And Outputs

Initialization of the 29C82, loading of the blocks to be decoded into the input FIFO (FIFE), and pixel read after decoding, are all accomplished over the microprocessor interface connected to an ADDR[3..0]/ DATA[7..0] demultiplexed bus. The 29C82 is a peripheral circuit of the microprocessor, it is accessed the 29C82 through a 16-address window which corresponds to registers, tables or FIFOs. A STATUS register, combined with an INTerrupt register, informs the µP on any detected change in status or errors.

A PIXEL[7..0] bus, dedicated to an image memory interface, is used to read the pixels after decompression. This bus is able to carry 8×8 pixel blocks (line by line) at a system clock frequency of 20 MHz. When the bus is not used (pixel output on the system bus), PIXEL (7..0] and CTYPE[1..0] are in the high-impedance mode in order to allow the connection of several 29C82's in parallel.

Figure 4. 29C82 Read/Write Bus Cycles

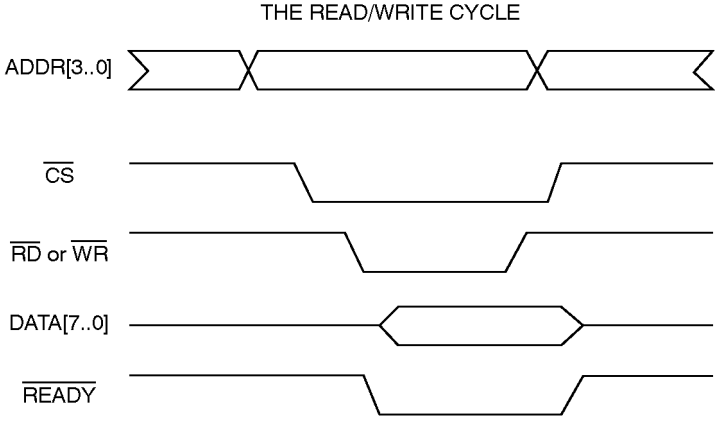
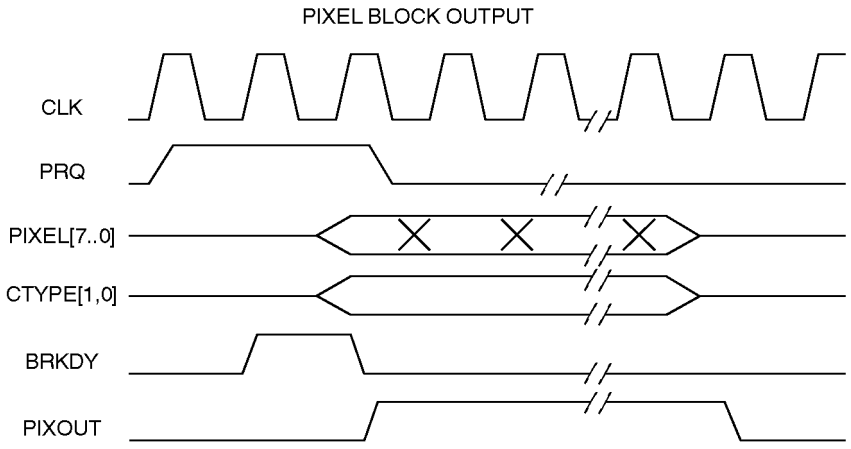


Figure 5. Pixel Block Output Timing



Description of the Registers

The microprocessor communicates with the 29C82 over an 8-bit data bus and a 4 bits address bus, (16 possible addresses). When one of these 16 addresses corresponds to a FIFO or a table or a multiple register, read or write access to the various parts of the FIFO, table or register set is accomplished sequentially, keeping the address on ADDR[3..0] constant.

The VLC Tables

The 29C82 contains 4 VLC tables. Two are for DC coefficients and two for AC coefficients. Each VLC table contains the HUFFMAN tree associated with decoding of the various AC and DC coefficients.

- the DC tables
  - address : T0_DC : 00H.
    T1_DC : 03H.
    depth : 32 × 5-bit words **centered** on the 5 LSB's of the byte written over the microprocessor bus. Access : write in sequential mode.
    T0_DC/T1_DC selection is performed in the NTAB register of the GENMUX register set.

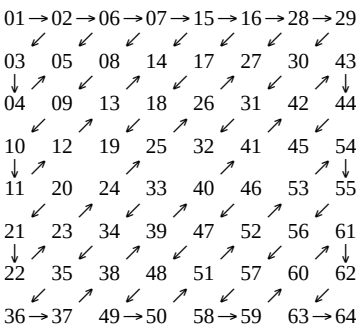
- the AC tables
  - address : T0_AC : 01H (Nodes)
    and 02H (leaves).
    T1_AC : 04H (Nodes)
    and 05H (leaves).
  - depth : 512 × 9-bit words, with the 9th bit (LSB) obtained by address decoding. This is forced to 1 for a leaf of the HUFFMAN tree, and to 0 for a node of the tree.
  - access : write in sequential mode.

The Quantization Tables

The 29C82 contains 4 quantization tables :

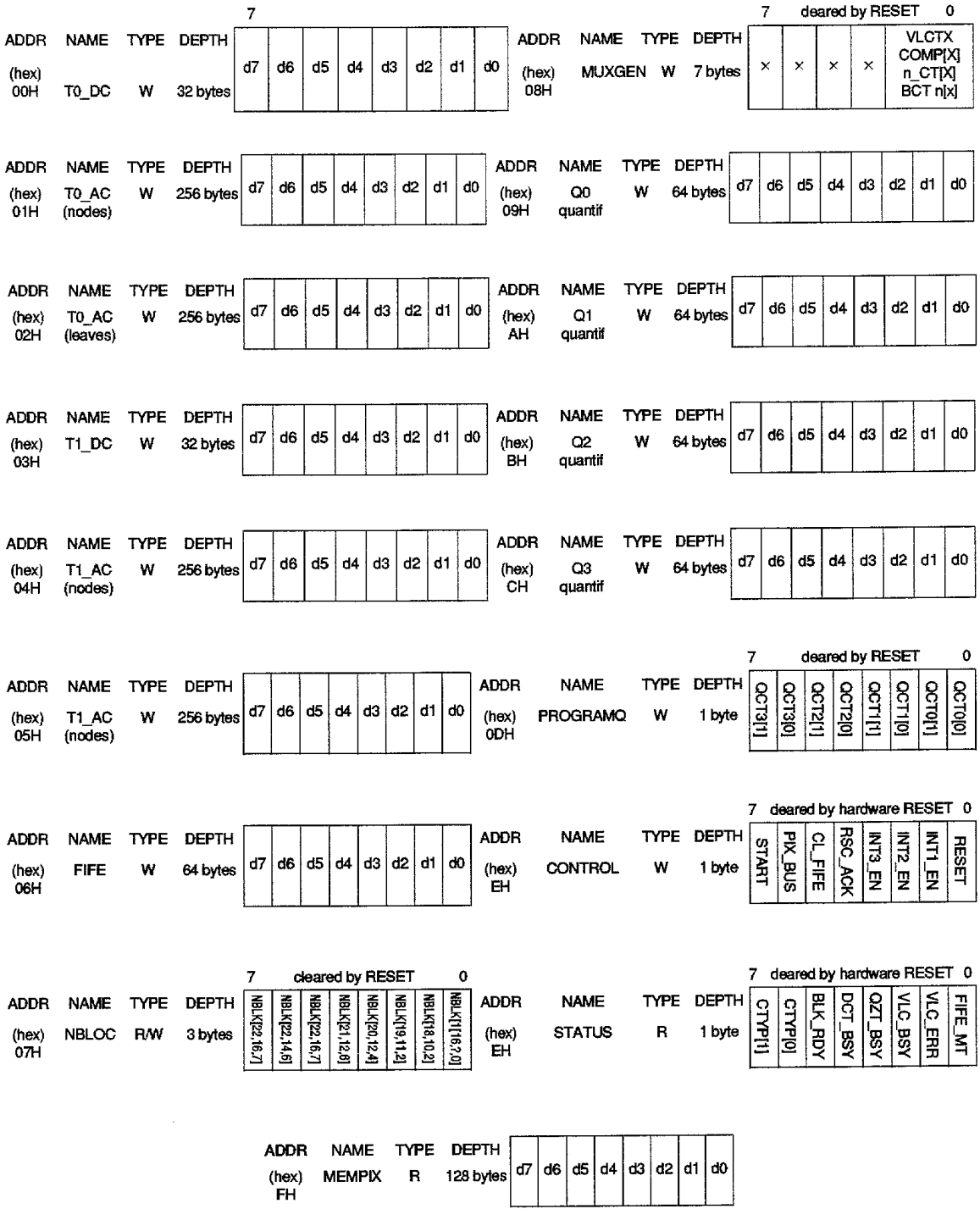
- address Q0 : 09H
  Q1 : 0AH
  Q2 : 0BH
  Q3 : 0CH
- depth : 64 × 8-bit words
- access : write in sequential mode.

The threshold levels in the quantization tables are written into Q0 to Q3 in the following order :



Q0 to Q3 selection is accomplished using the PROGRAMQ register.

Registers


ADDR NAME TYPE DEPTH
(hex) 00H T0_DC W 32 bytes
7 d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 08H MUXGEN W 7 bytes
7 deared by RESET 0
× × × × VLCTX COMP[X] n_CT[X] BCT n[x]
ADDR NAME TYPE DEPTH
(hex) 01H T0_AC (nodes) W 256 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 09H Q0 quantif W 64 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 02H T0_AC (leaves) W 256 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) AH Q1 quantif W 64 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 03H T1_DC W 32 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) BH Q2 quantif W 64 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 04H T1_AC (nodes) W 256 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) CH Q3 quantif W 64 bytes
d7 d6 d5 d4 d3 d2 d1 d0
ADDR NAME TYPE DEPTH
(hex) 05H T1_AC (nodes) W 256 bytes
d7 d6 d5 d4 d3 d2 d1 d0
7 deared by RESET 0
ADDR NAME TYPE DEPTH
(hex) 0DH PROGRAMQ W 1 byte
QCT3[1] QCT3[0] QCT2[1] QCT2[0] QCT1[1] QCT1[0] QCT0[1] QCT0[0]
ADDR NAME TYPE DEPTH
(hex) 06H FIFE W 64 bytes
d7 d6 d5 d4 d3 d2 d1 d0
7 deared by hardware RESET 0
ADDR NAME TYPE DEPTH
(hex) EH CONTROL W 1 byte
START PIX_BUS CL_FIFE RSC_ACK INT3_EN INT2_EN INT1_EN RESET
7 cleared by RESET 0
ADDR NAME TYPE DEPTH
(hex) 07H NBLOC R/W 3 bytes
NBLK[P22:16.7] NBLK[P22:14.6] NBLK[P22:16.7] NBLK[P21:12.6] NBLK[P20:12.4] NBLK[P19:11.2] NBLK[P18:10.2] NBLK[P17:16.7:0]
7 deared by hardware RESET 0
ADDR NAME TYPE DEPTH
(hex) EH STATUS R 1 byte
CTYP[1] CTYP[0] BLK_RDY DCT_BSY QZT_BSY VLC_BSY VLC_ERR FIFE_MT
ADDR NAME TYPE DEPTH
(hex) FH MEMPIX R 128 bytes
d7 d6 d5 d4 d3 d2 d1 d0

The FIFO's

The input FIFO - FIFE

- address : 06H
- depth : 64 bytes
- access type : sequential write
- pointer reset by hardware or software RESET

Write access to FIFE is made through the mP bus. Reading from FIFE is done via the VLC decoder. The FIFE_MT bit of the status register is used to indicate that FIFE is three-quarters empty (3 16 bytes available in FIFE). The microprocessor is responsible for limiting the number of write into the FIFO so that its capacity is not exceeded. When the FIFO is 100% empty, the VLC decoder does not hold up 29C82 operation.

The output FIFO - MEMPIX

- address : 0F
- depth : 64 bytes 2 y 64
- access type : sequential read
- pointer reset by RESET

Writing to the output Pixel FIFO (MEMPIX) is achieved using the DCT operator. Read is accomplished :

– over the PIXEL[7..0] bus if PIX_BUS = 1 (CONTROL register)
– over the DATA[7..0] bus if PIX_BUS = 0

The PIXELS FIFO is splitted into two pages of 64 bytes, accessed alternately by the DCT operator. While the operator is calculating the inverse DCT for a block of 8x8 coefficients, the second page of 64 bytes can be accessed over the PIXEL[7..0] bus or via the DATA[7..0] bus to output the results of the DCT calculation on the preceding 8x8 block. When the output FIFO is full up, the BLK_RDY bit of the STATUS register goes to 1, and an interrupt (INT) is activated if this has been enabled by the INT3_EN bit in the CONTROL register.

The registers

CONTROL address : 0EH Access type : write RESET Hardware

START	PIX_BUS	CL_FIFE	RSC_ACK	INT3_EN BLK_RDY	INT2_EN FIFE_MT	INT1_EN VLC_ERR	RESET SOFT
7	6	5	4	3	2	1	0

START : DECODING START.

START=1 enables the start of image decoding. This bit is reset to 0 one clock period (CLK) after it has been written to 1 in the control register.

PIX_BUS : BUS PIXEL

- PIX_BUS=0. Read access to the output FIFO, MEMPIX, occurs over the microprocessor bus. The PIXEL[7..0] and CTYPE[1..0] outputs are in the high-impedance state, PIXOUT=0, BLKRDY is active and PRQ is inactive.
- PIX_BUS=1, read access to the output FIFO is performed over the PIXEL[7..0] private bus.

CL_FIFE : Input FIFO reset
CL_FIFE=1 causes a zero reset of the input FIFO, FIFE. This bit is automatically reset to 0 one CLK period after it has been written to 1 in the CONTROL register.

RSC_ACK : DECODING OF THE RESYNCHRONISATION MARKERS.
RSC_ACK=1 enables decoding of the RSC0(FFD0) to RSC7(FFD7) markers.

Note : The JPEG standard specifies the use of re-synchronisation markers in order to segment the image. These markers are alined on the byte borders. They can be used for partial re-transmission of images after detection of errors.

The Interrupts

When the INTx_EN enable bit is at 1, the interrupt request (INT) can be activated :

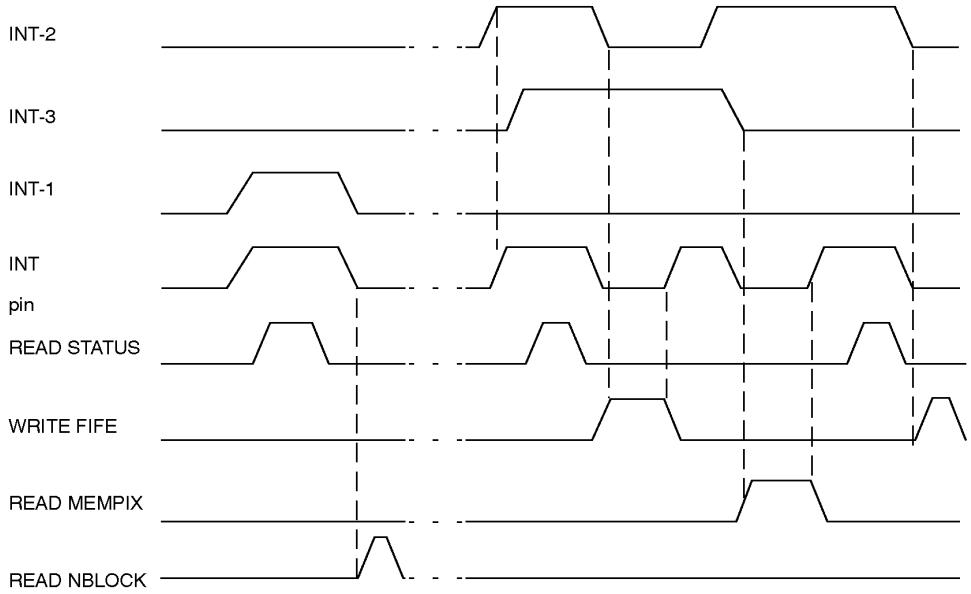
– INT3_EN : if the output FIFO, MEMPIX, contains a full block of 8x8pixels (bit BLK_RDY=1 in the STATUS register). Reading a single pixel from MEMPIX causes a reset of the interrupt request.
– INT2_EN : if the input FIFO, FIFE, is three-quarters empty (bit FIFE_MT=1 in the STATUS register).
– INT1_EN : when a VLC decoding error is detected (bit VLC_ERR=1 in the STATUS register).

The microprocessor must read the STATUS register to identify the cause of the interrupt. One interrupt process cycle allows the output INT to be forced to 0 (at least

during the cycle period) and the associated interrupt request to be de-activated in the following way :

– INT-1 (VLC error) : read NBLOCK register. If INT-2 or INT-3 are active, INT goes back to 1 after fully reading NBLOCK (3 bytes).
– INT-2 (FIFE three-quarters empty) : a write to FIFE is used to reset the FIFE_MT bit of the STATUS register to 0, and to force the INT output to 0. If INT-1 or INT-3 are active, INT goes back to 1 after the write cycle to FIFE. Caution : the fact that one or more bytes have been written to FIFE after an INT-2 interrupt request does not necessarily cause the FIFO pointer to go to a value over 16 (the 3/4 empty level). In such a case, a new INT-2 interrupt occurs.
– INT-3 (block ready in the output FIFO) : reading (one pixel only) from MEMPIX resets BLK_RDY (the status register) and the BLKROY/INT outputs to “0”. If INT-1 or INT-2 are active, INT goes back to 1 after the first read cycle from MEMPIX.

Figure 6.



INT3_EN : “OUTPUT BLOCK READY” INTERRUPT ENABLE. INT3_EN=1 enables transmission of INT-3 (BLOCK READY) interrupt request to the microprocessor.

INT2_EN : “FIFE 3/4 EMPTY” INTERRUPT ENABLE. INT2_EN=1 enables transmission of INT-2 (FIFE 3/4 EMPTY) interrupt request to the microprocessor.

INT1_EN : “VLC ERROR” INTERRUPT ENABLE. INT1_EN=1 enables transmission of INT-2 (VLC ERROR) interrupt request to the microprocessor.

RESET : SOFTWARE RESET RESET=1 is used to re-initialise the internal STATUS register and also the FIFO pointers without affecting the contents of the tables and the CONTROL, PROGRAMQ, NBLOCK, and MUXGEN registers. This bit is automatically reset to 0 one CLK period after it has been written to “1”.

STATUS address 0EH Access Type : read RESET SOFTWARE or HARDWARE

CTYP [1] 7	CTYP [0] 6	BLK_RDY 5	DCT_BSY RSC [2] 4	QZT_BSY RSC [1] 3	VLC_BSY RSC [0] 2	VLC_ERR 1	FIFE_MT 0

CTYP[1..0] : COMPONENT TYPE.
These two bits are used to indicate the image component to which the pixel block available in MEMPIX belongs. The CTYP[1..0] bits are in the same state as the CTYPE[1..0] outputs.

BLK_RDY : PIXEL BLOCK READY. The BLK_RDY bit goes to 1 when a new 8 × 8 pixel block is available in MEMPIX. BLK_RDY bit is in the same state as the BLKRDY output.

DCT_BSY : DCT OPERATOR ACTIVE (FOR INFORMATION ONLY).
DCT_BSY=1 indicates that the reverse DCT operator is active. If bit VLC_ERR=1, this bit is re-defined as RSC[2] (see description of the VLC_ERR bit).

QZT_BSY : DE-QUANTIZATION OPERATOR ACTIVE (FOR INFORMATION ONLY). QZT_BSY=1 indicates that the de-quantization operator is active. If bit VLC_ERR=1, this bit is re-defined as RSC[1] (see description of the VLC_ERR bit).

VLC_BSY : VLC DECODING OPERATOR ACTIVE (FOR INFORMATION ONLY).
VLC_BSY=1 indicates that the VLC operator is active. If bit VLC_ERR=1, this bit is re-defined as RSC[0] (see description of the VLC_ERR bit). The VLC operator is always active, even if FIFE is completely empty. VLC_BSY is reset to 0 only after a RESET and before the START bit has been set to 1.

VLC_ERR : VLC ERROR. VLC_ERR=1 indicates that a VLC decoding error has been detected. This type of error can be caused by :
– detection of an EOI or RSC marker with the content of the NBLOCK register different from 0.
– detection of a VLC code with a length of more than 16 bits.
– a coefficient pointer over 63.

In this case, the RSC[2..0] of the status register represent the three LSB's of the last re-synchronisation marker detected before the error. Output INT goes to 1 if bit INT1_EN of the CONTROL register is at 1. The microprossor reads from NBLOCK the number of blocks remaining to be decoded (NBLOCK acts as a down-counter) in order to reset the interrupt and the VLC_ERR bit to 0. In the case where there is no re-synchronisation, the µP (APPLICATION) selects the strategy to be used (re-transmission of BLOCK, SCAN, FRAME, IMAGE, etc.) When one or more re-synchronisation markers exist, the 29C82 waits for the next resynchronisation marker so that it can continue decoding of the current image.

FIFE_MT : FIFE 3/4 EMPTY.
FIFE_MT=1 when the input FIFO, FIFE, is three-quarters empty (16 bytes left ≤). When the INT2_EN bit of the CONTROL register is at 1, the INT output goes to 1. In this case, the microprossor can write 48 new bytes into FIFE.

PROGRAMQ address 0DH Access type : write Reset : HARDWARE

QCT3 [1] 7	QCT3 [1] 6	QCT2 [1] 5	QCT2 [0] 4	QCT1 [1] 3	QCT1 [0] 2	QCT0 [1] 1	QCT0 [0] 0

QCT3[1..0] : "QUANTIZATION TABLE" REFERENCE FOR COMPONENT 3.
These two bits code the reference of the quantization table for a block which belongs to a type-3 image component.

QCT2[1..0] : "QUANTIZATION TABLE" REFERENCE FOR COMPONENT 2.
These two bits code the reference of the quantization table for a block which belongs to a type-2 image component.

QCT1[1..0] : "QUANTIZATION TABLE" REFERENCE FOR COMPONENT 1.
These two bits code the reference of the quantization table for a block which belongs to a type-1 image component.

QCT0[1..0] : "QUANTIZATION TABLE" REFERENCE FOR COMPONENT 0.
These two bits code the reference of the quantization table for a block which belongs to a type-0 image component.

The Multiple Registers

NBLOCK address 07H Access type : sequential read/write
depth : 3 bytes RESET HARDWARE

NBLK[23] 7	NBLK[22] 6	NBLK[21] 5	NBLK[20] 4	NBLK[19] 3	NBLK[18] 2	NBLK[17] 1	NBLK[16] 0
NBLK[15] 7	NBLK[14] 6	NBLK[13] 5	NBLK[12] 4	NBLK[11] 3	NBLK[10] 2	NBLK[09] 1	NBLK[08] 0
NBLK[07] 7	NBLK[06] 6	NBLK[05] 5	NBLK[04] 4	NBLK[03] 3	NBLK[02] 2	NBLK[01] 1	NBLK[00] 0

NBLK[23..0] : NUMBER OF BLOCKS
NBLK[23..0] contains the number of blocks to be decoded between the two markers (RSC_ACK=1) or for a full image (RSC_ACK=0). The 29C82 uses NBLOCK as a down-counter.

MUXGEN address 08H Access type : sequential write depth : 7 bytes RESET HARDWARE

The ADCT algorithm allows two interlacing modes :

– the frame interlace mode, where each component is coded separately and transmitted sequentially for a complete image (e.g. press images, “yellow”, “blue”, “red”, “black” transmission).

The block interlacing mode, where the 8x8 blocks are interlaced in accordance with a repeating pattern (MDU) (e.g. Videotext transmission of two 8x8 luminance blocks, one 8 × 8 “red” chrominance block, 1 “blue” chrominance block, ...)

MUXGEN contains 7 registers :

NTAB

X 7	X 6	X 5	X 4	VLCT3 3	VLCT2 2	VLCT1 1	VLCT0 0

VLCT3 : VLC TABLE FOR TYPE-3 IMAGE COMPONENT.
This bit indicates which VCL table will be used for a block belonging to the type-3 component. (“0” for T0_DC/ T0_AC and ”1” for T1_DC/T1_AC).

VLCT2 : VLC TABLE FOR TYPE-2 IMAGE COMPONENT.
This bit indicates which VCL table will be used for a block belonging to the type-2 component. (“0” for T0_DC/ T0_AC and “1” for T1_DC/T1_AC).

VLCT1 : VLC TABLE FOR TYPE-1 IMAGE COMPONENT.
This bit indicates which VCL table will be used for a block belonging to the type-1 component. (“0” for T0_DC/T0_AC and “1” for T1_DC/T1_AC).

VLCT0 : VLC TABLE FOR TYPE-0 IMAGE COMPONENT.
This bit indicates which VCL table will be used for a block belonging to the type-0 component. (“0” for T0_DC/T0_AC and “1” for T1_DC/ T1_AC).

NCOMP

X 7	X 6	X 5	X 4	X 3	X 2	COMP[1] 1	COMP[0] 0

COMP[1..0] : NUMBER OF IMAGE COMPONENTS
COMP[1..0] contains the number of image components to be decoded :
11 → 4 components
10 → 3 components
01 → 2 components
00 → 1 components

ORDER

1_CT[1]	1_CT[0]	2_CT[1]	2_CT[0]	3_CT[1]	3_CT[0]	4_CT[1]	4_CT[0]
7	6	5	4	3	2	1	0

The “quantization table”, “VLC table”..., data associated with each image component are clearly specified in the JPEG header (see appendix 2). The ORDER register is used to associate indirectly a component type with a component of the JPEG frame. The PROGRAMQ and NTAB registers are used to associate the VLC and quantization tables with each component type.

1_CT[1..0] : TYPE OF FIRST COMPONENT. These two bits code the type of the first component to be processed.

2_CT[1..0] : TYPE OF FIRST COMPONENT. These two bits code the type of the second component to be processed.

3_CT[1..0] : TYPE OF FIRST COMPONENT. These two bits code the type of the third component to be processed.

4_CT[1..0] : TYPE OF FIRST COMPONENT. These two bits code the type of the fourth component to be processed.

BLOCK-0

BCT0[7]	BCT0[6]	BCT0[5]	BCT0[4]	BCT0[3]	BCT0[2]	BCT0[1]	BCT0[0]
7	6	5	4	3	2	1	0

BCT0[7..0] : NUMBER OF SUCCESSIVE BLOCKS FOR TYPE-0 COMPONENT.

The BLOCK-0 register indicates (in 8 bits) the number of successive 8 × 8 blocks contained in 1 MDU.

BLOCK-1

BCT1[7]	BCT1[6]	BCT1[5]	BCT1[4]	BCT1[3]	BCT1[2]	BCT1[1]	BCT1[0]
7	6	5	4	3	2	1	0

BCT1[7..0] : NUMBER OF SUCCESSIVE BLOCKS FOR TYPE-1 COMPONENT.

The BLOCK-1 register indicates (in 8 bits) the number of successive 8 × 8 blocks contained in 1 MDU.

BLOCK-2

BCT2[7]	BCT2[6]	BCT2[5]	BCT2[4]	BCT2[3]	BCT2[2]	BCT2[1]	BCT2[0]
7	6	5	4	3	2	1	0

BCT2[7..0] : NUMBER OF SUCCESSIVE BLOCKS FOR TYPE-2 COMPONENT.

The BLOCK-2 register indicates (in 8 bits) the number of successive 8 × 8 blocks contained in 1 MDU.

BLOCK-3

BCT3[7]	BCT3[6]	BCT3[5]	BCT3[4]	BCT3[3]	BCT3[2]	BCT3[1]	BCT3[0]
7	6	5	4	3	2	1	0

BCT3[7..0] : NUMBER OF SUCCESSIVE BLOCKS FOR TYPE-3 COMPONENT.

The BLOCK-3 register indicates (in 8 bits) the number of successive 8 × 8 blocks contained in 1 MDU.

Initialisation

Before it can be used, the 29C82 has to be programmed by the microprocessor as follows :

1 - the RESET bit of the CONTROL register is put to 1 for a partial reset, or a complete reset is achieved by forcing the RESET pin to 1.

2 - in any order :
- load the NBLOCK register
- load the MUXGEN register
- load the VLC table (example in appendix 4)
- load the quantization tables (example in appendix 5)
- write to register PROGRAMQ
- write to the CONTROL register (START=0).

3 - Put the START bit of the CONTROL register to 1.

4 - Load the input FIFO (FIFE). This step can be programmed after setting CL_FIFE and before setting the START bits.

After the 29C82 has started decoding, the microprocessor is able to poll the STATUS register or the state of the output INT to discover the state of progress of the current decoded block.

Electrical Characteristics

Absolute Maximum Ratings

VCC to earth . . . . . –0.5V to +7V
Input/output voltage . . . . . –0.3V to VCC + 0.3V
Storage temperature . . . . . –65 to +150°C

Operating Characteristics

VCC supply voltage . . . . . 4.5V to +5.5V
Temperature . . . . . 0 to 70°C
Load capacity . . . . . 50pF on each output

Electrical DC Characteristics

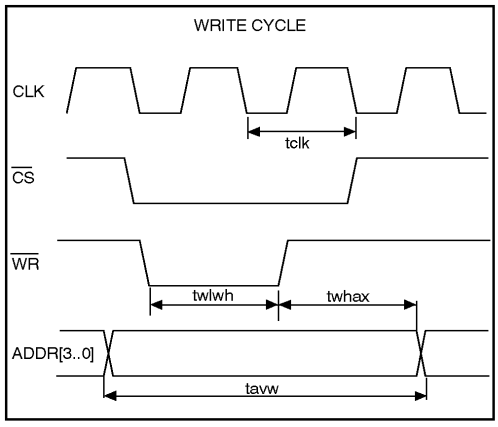
PARAMETER	CONDITIONS	MIN	MAX
Input voltage VIL	I=5µA		0.8V
Input voltage VIH	I=5µA	2.2V	
Input current leakage: All inputs except Test 1 and Test 2		– 5µA	+ 5µA
Input current leakage: Test 1 and Test 2			+ 100µA
Output voltage, level 0	I = 6.4mA		0.4V
Output voltage, level 1	6.4mA	2.4 V	
Dynamic consumption	VCC=5V, 20MHz		120mA

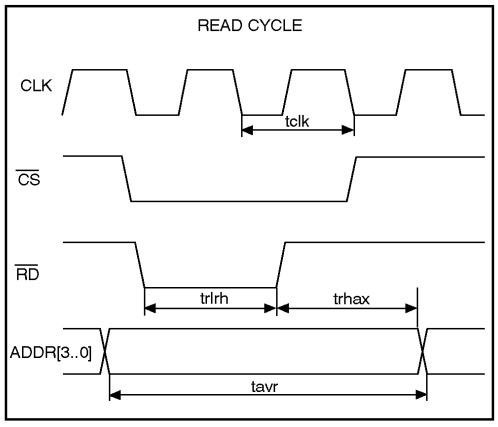
AC Timings

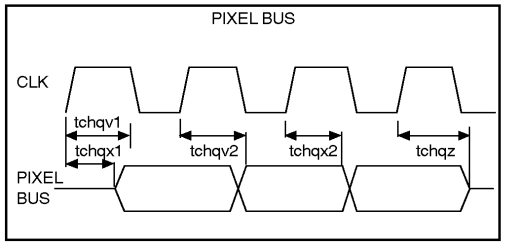
VCC = 5V ± 10%. Temperature = 0 to 70°C
50 pF load on all outputs

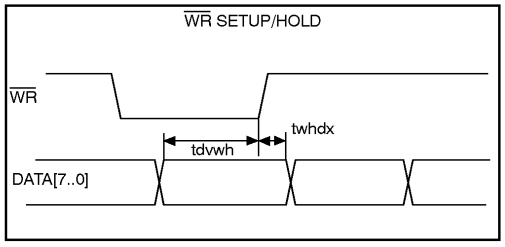
NAME	DESCRIPTION	MIN ns	MAX ns
tphch	PRQ to CLK setup time	20	
tchpl	PRQ to CLK hold time	5	
tchqv	CLK to PIXEL_BUS data delay		40
tchqx	CLK to PIXEL_BUS not floating	5	
tchqz	CLK to PIXEL_BUS floating		35
tdvwh	DATA[0..7] to WR setup time	5	
twhdx	DATA[0..7] to WR hold time	5	
twhyh	CS/WR high to READY low delay		20
twlyl	CS/WR low to READY low delay		20
tchcl	minimum high CLK pulse width	25	
tclch	minimum low CLK pulse width	25	
trlyl	CS/RD low to READY low		5tclk+30
trhyh	CS/RD high to READY high		25
trldv	RD/CS to DATA_BUS data delay		5tclk+35
trldx	RD/CS to DATA_BUS not floating		5tclk+5
trhdz	RD/CS high to DATA_BUS floating		20
tclk	CLK period	50	
twlwh	write pulse width	1tclk	
twhax	ADDR[3..0] / $\overline{WR}$ hold time	2tclk	
tavw	address hold time (write)	3tclk	
trlrh	$\overline{RD}$ pulse width	6tclk	
trhax	ADDR[3..0] / $\overline{RD}$ hold time	1tclk	
tavr	address hold time (read)	7tclk	

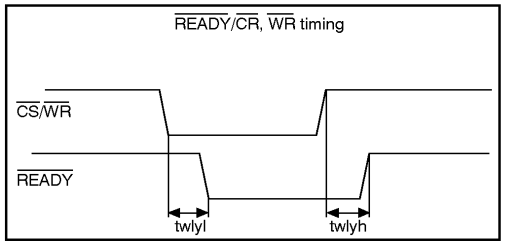
AC Timings

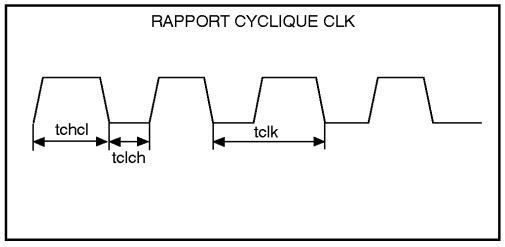

WRITE CYCLE
CLK
tclk
CS
WR
twlwh
twhax
ADDR[3..0]
tavw

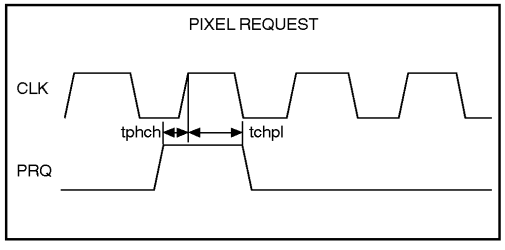

READ CYCLE
CLK
tclk
CS
RD
trlrh
trhax
ADDR[3..0]
tavr

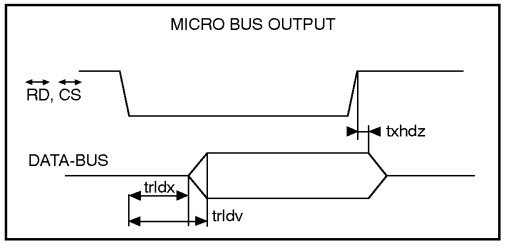

PIXEL BUS
CLK
tchqv1
tchqx1
tchqv2
tchqx2
tchqz
PIXEL BUS

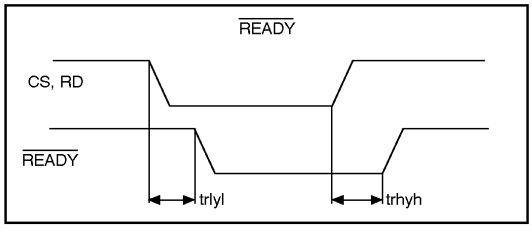

WR SETUP/HOLD
WR
twhdx
tdvwh
DATA[7..0]


READY/CR, WR timing
CS/WR
READY
twlyl
twlyh


RAPPORT CYCLIQUE CLK
tchcl
tclch
tclk


PIXEL REQUEST
CLK
tphch
tchpl
PRQ

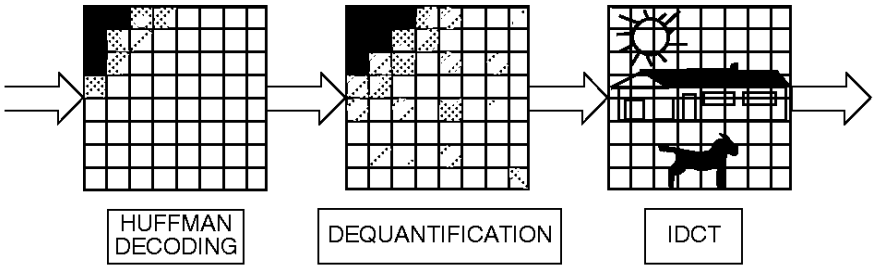

MICRO BUS OUTPUT
RD, CS
txhdz
DATA-BUS
trldx
trldv


READY
CS, RD
READY
trlyl
trhyh

Appendix 1

The various stages of ADCT

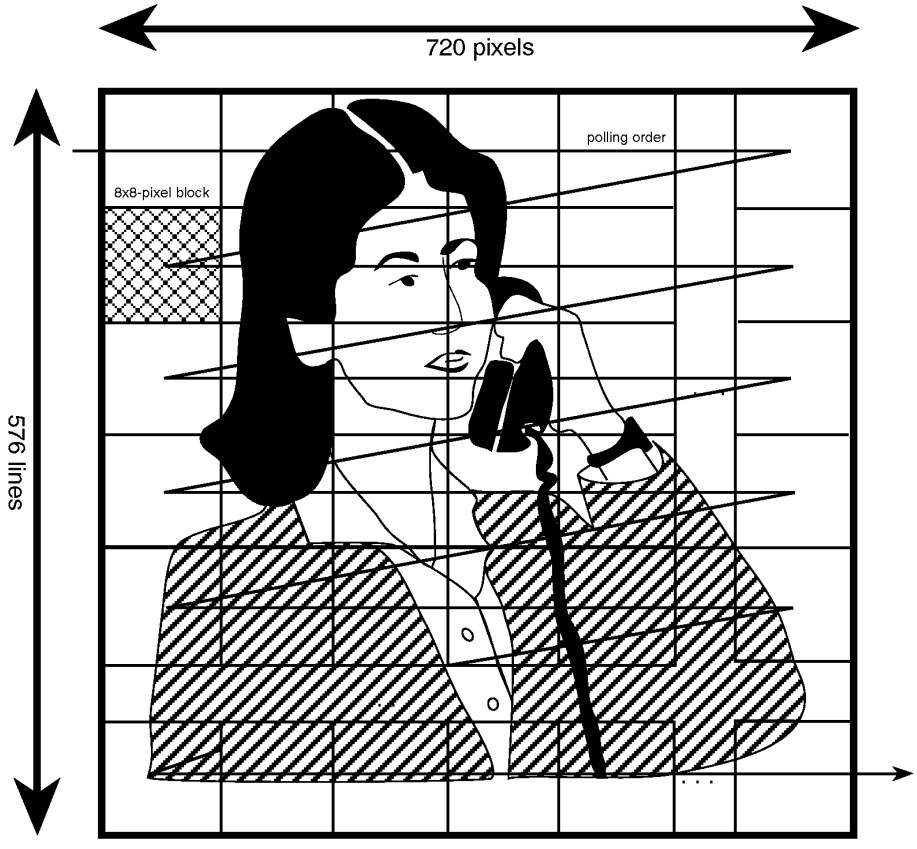
The successive stages of image de-compression for a basic JPEG system are the VLC coding (Huffman) decoding, the dequantization and the Inverse cosine transform.



The two-dimensional Cosine Transform

The image is divided into 8 × 8 pixel blocks and then polled in the order shown in figure 1. In the case of a 720 × 576 pixel image in 4:2:2 for example, the image will be divided into 90 × 72 blocks for Y luminance and 45 × 72 blocks for Cr and Cb chrominance.

Figure 7.



Level transposition

Before applying the Cosine Transform to the unsigned data (Y,Cr,Cb) a level transposition is performed by subtracting 128 (for 8-bit data).

The Cosine Transform

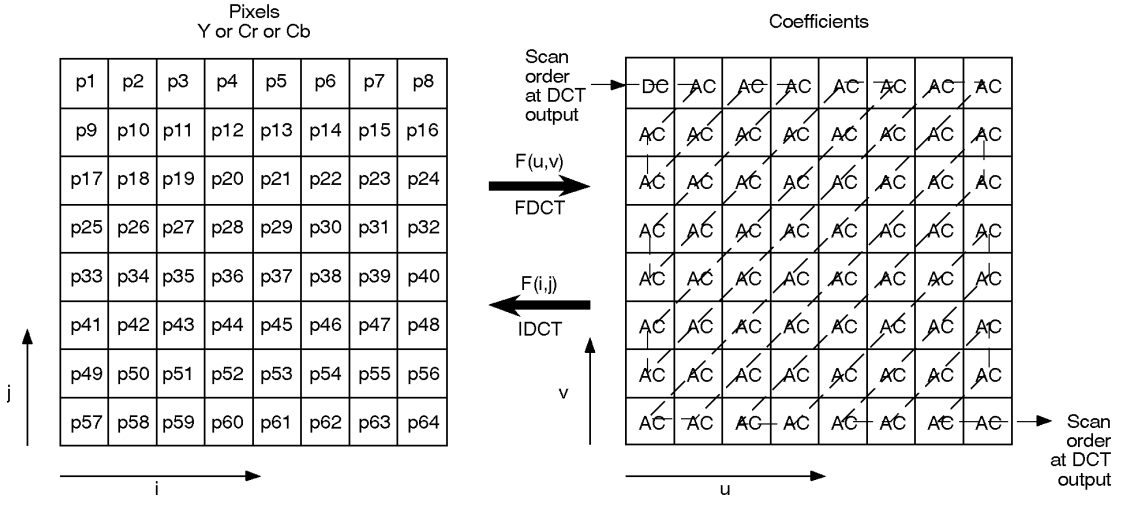
For each block, the Cosine Transform is used to convert from the space domain (the pixels) to the frequency domain (the coefficients), by applying the direct Cosine Transform (FDCT).

$$F(u,v) = 1/4C(u) \cdot C(v) \sum_{i=0}^{7} \sum_{j=0}^{7} F(i,j) \cos(2i + 1)u \cdot \pi/16 \cdot \cos(2j + 1)v \cdot \pi i/16$$

or from the frequency domain to the space domain using the inverse Cosine Transform (IDCT).

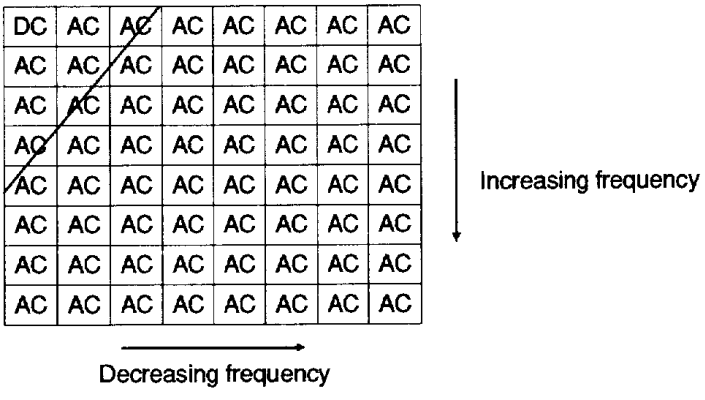
$$F(i,j) = 1/4 \sum_{u=0}^{7} \sum_{v=0}^{7} C(u) \cdot C(v) F(u,v) \cos(2i + 1)u \cdot \pi/16 \cdot \cos(2j + 1)v\pi i/16$$

Figure 8.



Quantization

An essential property of the Cosine Transform is the concentration of the energy in the coefficients between each 8x8 block into the DC component and some 'low-frequency" AC components.



Since the human eye is not very sensitive to high-frequency AC components, these can be quantized in a coarser manner.

Each coefficient is quantized as follows :

F(u,v)0 : C(u,v) =
full division (F(u,v)+(Q(u,v)/2))/Q(u,v)
F(u,v) : C(u,v) =
full division (F(u,v) – (Q(u,v)/2))/Q(u,v)

F(u,v) is the coefficient before quantization
C(u,v) is the coefficient after quantization
Q(u,v) indicates no quantization

De-quantization is performed in a similar fashion :

$$F'(u,v) = C(u,v) * Q(u,v)$$

HUFFMAN coding

The average number of bits necessary to code a signal without loss of information can be measured by entropy of the signal. A procedure used commonly to construct the associated variable-length code words is known as HUFFMAN coding.

One intuitively associates short code words to frequent signal values and longer code words to the less frequent values.

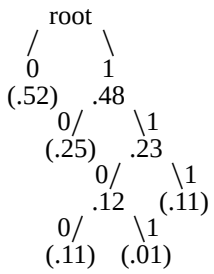
Example :

signal value	probability
0	.11
1	.52
2	.25
3	.01
4	.11

a - classification of probabilities in decreasing order, and grouping of the lowest probability values.

.52, .25, .11, .11, .01
.52, .25, .12, .11,
.52, .25, .23
.52, .48

b - construction of the HUFFMAN tree



c - association of Code

SIGNAL VALUE	PROBABILITY	HUFFMAN CODE	CODE LENGTH
1	.52	0	1
2	.25	10	2
0	.11	111	3
4	.11	1100	4
3	.01	1101	4

This technique is used in the JPEG frame for coding the DC and AC components after quantization. Each 8 × 8 block of coefficients is transmitted in the following order :

0	1	5	6	14	15	27	28	0 =DC coefficient
2	4	7	13	16	26	29	42	
3	8	12	17	25	30	41	43	
9	11	18	24	31	40	44	53	
10	19	23	32	39	45	52	54	
20	22	33	38	46	51	55	61	
35	36	48	49	57	58	62	63	

Appendix 2

Structure of a JPEG frame

A JPEG image is divided into signalling or data segments. Each segment starts with an FFxx marker in hex notation. The last xx byte of the marker identifies the function of the segment. The last data item of a segment is ended with a “1” so that FFxx is aligned on a byte border.

Figure 9.

S0I	FFD8	IMAGE START
S0C	FFFE	COMMENTS
DQT	FFD8	QUANTIZATION TABLES
DHT	FFC4	VLC TABLES
S0F0	FFC0	FRAME Parameter definition : sample precisio,n number of line,s number of samples per lin,e number of components in fram,e vertical sampling factor horizontal sampling facto,r selection of quantization table.
S0S	FFDA	SCAN 1 Parameter definition : number of scan components, choice of HUFFMAN matrix (AC, DC), choice of mode (sequential, progressive). Coded data
E0I	FFD9	IMAGE END

Comments

Example

fffe : COM marker
0020 : Comment length (30 usable bytes)
432d 4355 4245 204d : MATRA MHS Electronic – APPLICATION LAB
6963 726f 7379 7374 : 1992
656d 7320 496e 632e : JPEG image example
2031 2e30 3000

Quantization tables

The quantization tables are always included in the header of the frame (1 per component type).

Example

ffdb : DQT marker
007F : Table length (2 tables × 8 bits)
00 : Table 0 : luminance
10 0b 0c 0e 0a 10 0e
0d 0e 12 11 10 13 18 28
1a 18 16 16 18 31 23 25
1d 28 3a 33 3c 3c 39 33
38 37 40 48 5c 4e 40 44
57 45 37 38 50 6d 51 57
5f 70 64 78 5c 65 67 63

01 : table 1 : Chrominance (Cr/Cb)
11 12 12 18 15 18 2f 1a
1a 2f 63 42 38 42 63 63
63 63 63 63 63 63 63 63
63 63 63 63 63 63 63 63
63 63 63 63 63 63 63 63
63 63 63 63 63 63 63 63
63 63 63 63 63 63 63 63
63 63 63 63 63 63 63 63

The HUFFMAN Tables

Coding the DC coefficients

Only the DC coefficient of the first block is transmitted in the form of an absolute value. The following coefficients are transmitted differentially in relation to this first block. The value of the DC coefficient to be transmitted is divided into 16 categories coded in 4 bits, ssss, and corresponds to the length of the DC delta code to be sent :

ssss	DC delta
0	0
1	–1,+1
2	–3,–2,2,3
3	–7..–4,4..7
4	–15..–8,8..15
5	–31..–16,16..31
6	–63..–32,32..63
7	–127..–64,64..127
8	–255..–128,128..255
9	–511..–256,256..511
10	–1023..–512,512..1023
11	–2047..–1024,1024..2047
12	–4095..–2048,2048..4095
13	–8191..–4096,4096..8191
14	–16383..–8192,8192..16383
15	–32767..–16384,16384..32767

The DC coefficients are coded in the following manner :

HUF(ssss)	DC coefficient

HUF(ssss) = the HUFFMAN code associated with ssss

The HUFFMAN table (HUFCODE) associated with ssss is not transmitted directly in the JPEG image header. In fact two tables, [BITS] and [HUFVAL], are used for this :

[BITS] is 16 bytes long and contains the number of ssss values (categories) coded in 1 bit, 2 bits, 3 bits ... 16 bits in the HUFFMAN table.

[HUFVAL] contains the list of ssss values (categories) sorted into ascending order of length of the HUFFMAN code (1 bit to 16 bits).

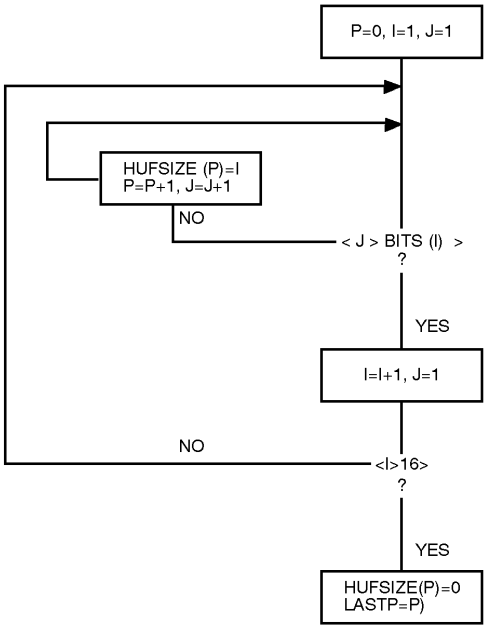
In a basic system, there can be a maximum of two [BITS] - [HUFVAL] pairs, transmitted by the DC coefficients.

Example :

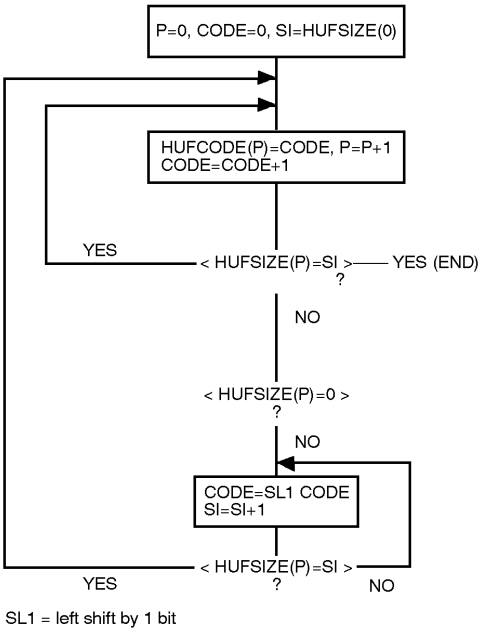
ffc4 : DHT marker
01a2 : Table length
00 : DC table 1
0001 0501 0101 0101 : BITS matrix
00100 0000 0000 0000
0001 0203 0405 0607 : HUFVAL matrix
0809 0a0b (12 elements)
01 : DC table 2
0003 0101 0101 0101 : BITS matrix
0101 0100 0000 0000
0001 0203 0405 0607 : HUFVAL matrix
0809 0a0b (12 elements)

Creation of matrices [EHUFCO] and [EHUFSI] from [BITS] and [HUFVAL] (JPEG standard) :

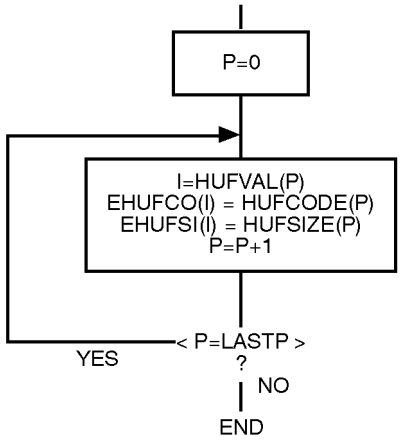
The following algorithm is used to create the [HUFSIZE] table containing the length of the HUFFMAN codes associated with ssss as a function of decreasing priority values :



The following algorithm is used to create the [HUFCODE] table containing the HUFFMAN codes associated with ssss as a function of decreasing priority values :



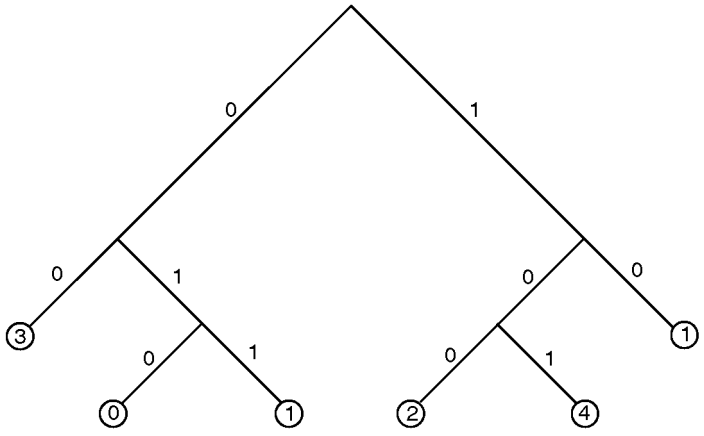
The following algorithm is used to reorganise HUFSIZE and HUFCODE as a function of decreasing ssss (HUFVAL) values :



Example

ssss	EHUHSI	EHUFCO
0	3	010
1	3	011
2	3	100
3	2	00
4	3	101
...	...	...
11	9	111111110

The reconstitution of the HUFFMAN decoding tree (see appendix 1) can be accomplished using EHUFCO and EHUFSI :



Coding the AC coefficients

The AC coefficients to be transmitted are divided into 256 categories, coded in 8 bits, nnnnssss.

nnnn gives the number of coefficients equal to 0 between two non-zero coefficients, and ssss gives the amplitude category for the AC coefficients.

00000000 represents E0B which is the end-of-block indicator.

ssss	AC coefficient
0	0
1	–1,+1
2	–3,–2,2,3
3	–7..–4,4..7
4	–15..–8,8..15
5	–31..–16,16..31
6	–63..–32,32..63
7	–127..–64,64..127
8	–255..–128,128..255
9	–511..–256,256..511
10	–1023..–512,512..1023
11	–2047..–1024,1024..2047
12	–4095..–2048,2048..4095
13	–8191..–4096,4096..8191
14	–16383..–8192,8192..16383
15	–32767..–16384,16384..32767

The AC coefficients are coded in the following manner :

HUF(nnnssss)	AC coefficient

HUF(nnnnssss) = the HUFFMAN code associated with nnnnssss

The HUFFMAN table (HUFCODE) associated with nnnnssss is not transmitted directly in the JPEG image header (ss DC coefficients). In fact two tables, [BITS] and [HUFVAL], are used for this.

In 16 bytes, [BITS] contains the number of nnnnssss values (categories) coded in 1 bit, 2 bits, 3 bits ... 16 bits in the HUFFMAN table.

[HUFVAL] contains the list of nnnnssss values (categories) sorted into ascending order of length of the HUFFMAN code (1 bit to 16 bits).

In a basic system, and for the AC coefificients, there can be a maximum of two [BITS] - [HUFVAL] pairs.

Example

```
10                          : AC table 1
0002 0103 0302 0403         : BITS matrix
0605 0404 0000 017d
0102 0300 0411 0612         : HUFVAL matrix
2131 4106 1351 6107         (162 elements)
2271 1432 8191 a108
2342 b1c1 1552 d1f0
2433 6272 8209 0a16
1718 191a 2526 2728
292a 3435 3637 3839
3a43 4445 4647 4849
4a53 5455 5657 5859
5a63 6465 6667 6869
6a73 7475 7677 7879
7a83 8485 8687 8889
8a92 9394 9596 9798
999a a2a3 a4a5 a6a7
a8a9 aab2 b3b4 b5b6
b7b8 b9ba c2c3 c4c5
c6c7 c8c9 cad2 d3d4
d5d6 d7d8 d9da e1e2
e3e4 e5e6 e7e8 e9ea
f1f2 f3f4 f5f6 f7f8
f9fa
11                          : AC table 2
0002 0102 0404 0304         : BITS matrix
0705 0404 0000 0277
0001 0203 1104 0521         : HUFVAL matrix
3106 1241 5107 6171         (289 elements)
1322 3281 0814 4291
a1b1 c109 2333 52f0
1562 72d1 0a16 2434
e125 f117 1819 1a26
2728 292a 3536 3738
393a 4344 4546 4748
494a 5354 5556 5758
595a 6364 6566 6768
696a 7374 7576 7778
797a 8283 8485 8687
8889 8a92 9394 9596
9798 999a a2a3 a4a5
a6a7 a8a9 aab2 b3b4
b5b6 b7b8 b9ba c2c3
c4c5 c6c7 c8c9 cad2
d3d4 d5d6 d7d8 d9da
e2e3 e4e5 e6e7 e8e9
eaf2 f3f4 f5f6 f7f8
f9fa
```

The algorithms given in the paragraph dealing with coding of the DCcoefficients are used to create the [EHUFSI] table containing the length of the HUFFMAN codes associated with nnnnssss and the [EHUFCO] table containing the HUFFMAN codes associated with nnnnssss.

Start of frame (DCT basic mode)

The number of components in a frame is limited to 4 (quadrichromy - black, blue, yellow and red). Each frame can contain several successive scans, each corresponding to one component of the image.

There are two scan modes :

– the non-interleave mode, where each scan contains a single image component,
– the interleavce mode, where a single scan contains the data for all components in the frame.

Within the frame, data is transmitted in MDU form. In the non-interlace mode, an MDU is a block of 8×8 coefficients which correspond, for example, to Y (luminance) or Cr/Cb (chrominance). In the interlace mode, and MDU is an interlacing of blocks defined by the sampling factor Hi,Vi and by NS, the number of components in the scan. In the following example, the scan is described by three components, Y, Cr and Cb, with respective sampling factors of 2h :1v, 1h :1v, and 1h :1v.

The MDU is described by the following :

$Y_1,Y_2,C_{r1},C_{b1};Y_3,Y_4,C_{r2},C_{b2}...Y_{2n},Y_{2n+1},C_{rn},C_{bn}...$

Data

The data are sent with the most significant byte leading and most significant bit leading.

Example :
93fd 1639 9954 7c9e
5fda 653e 667e fe57
3cee 1cee 2df8 ff00 when FF is created during coding, byte 00 is added to the data
.....
d476 96d1 cd74 e2df
7e0b 7ef3 f7ca bb63
c705 7e99 1c00 f418
ffds : marker EOI, end of image

Example :

FFC0	: SOF0 marker
0011	: fields length = 17
08	: sample precision = 8 bits
0100	: number of Y lines = 256
0100	: number of × samples per line = 256
03	: number of image components in the frame = 3
01	: component 1
2	: H1 horizontal sampling factor = 2
1	: V1 vertical sampling factor = 1
00	: selection of quantization table 0
02	: component 2
1	: H2 horizontal sampling factor = 1
1	: V2 vertical sampling factor = 1
01	: selection of quantization table 1
03	: component 3
1	: H3 horizontal sampling factor = 1
1	: V3 vertical sampling factor = 1
01	: selection of quantization table 1
Scanning	
ffda	: S0S = start scan marker
000C	: length of field (10 usable bytes)
03	: Ns = number of components in scan = 3
01	: selection of scan component = 1
0	: selection of HUFFMAN table, DC = 0
0	: selection of HUFFMAN table, AC = 0
02	: selection of scan component = 2
1	: selection of HUFFMAN table, DC = 1
1	: selection of HUFFMAN table, AC = 1
03	: selection of scan component = 3
1	: selection of HUFFMAN table, DC = 1
1	: selection of HUFFMAN table, AC = 1
003F00	: DCT/Sequential mode, Ss = 0, Se = 63, Ah = 0, Al = 0

Appendix 3

Operation of the VLC Decoder (29C82)

The VLC RAM contains the HUFFMAN decoding tree (see appendix 1). This tree is obtained from the [BITS] and [HUFVAL] matrices contained in the frame header (see appendix 2). The following example illustrates the extraction of DC and AC parameters from the data of the JPEG frame.

a - LOADING THE TABLES

Tables T0-DC and T0-AC are respectively programmed with :

T0-DC :

02	01	09	07	03	01	01	01	01	01	01	01	01	01	01	01
04	05	06	08	01	01	01	01	01	01	01	01	01	01	01	01

T0-AC

002	001	00E	008	00A	005	023	0E3	043	019	143	083	0A3	001	001	001
														.....	
														(16 lines)	
004	003	006	012	00c	007	025	010	063	014	016	001	0c3	001	001	001
														.....	
														(16 lines)	

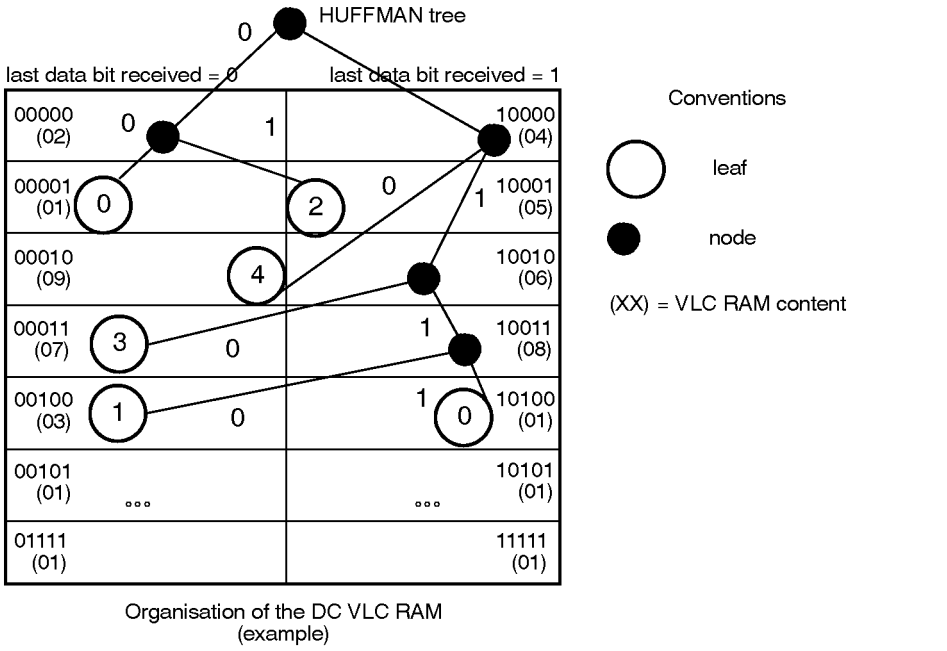
Each VLC table is divided into 2 zones (2×16 bytes for the DC tables and 2×256 bytes for the AC tables). The last bit extracted from the data stream indicated the zone to be addressed.

Conventions

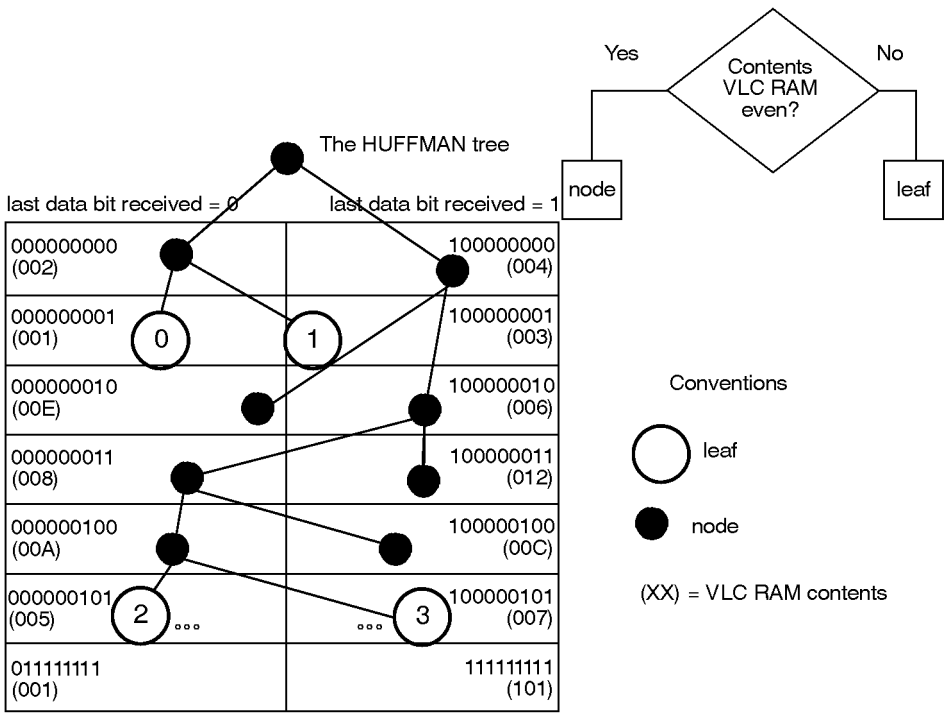
Each word in the VLC table containing a node of the HUFFMAN table is forced to an even value. In this case it contains the address of the next node (or leaf) of the HUFFMAN tree, shifted 1 bit left, the LSB being forced to 0.

Each word of the VLC table containing a leaf of the tree will be forced to an odd number. In this case it contains ssss (or nnnnssss) shifted 1 bit left, the LSB being forced to 1.

Figure 10.



Organisation of the DC VLC RAM
(example)



Organisation of the AC VLC RAM
(example)

b - DECODING THE DATA

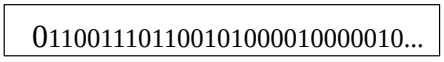
b1 - extraction of the DC parameters

Each DC VLC table is composed of 32 5-bit words, with the LSB forced to 0 or 1 depending on whether the word corresponds to a node or a leaf of the VLC tree.

Extraction of DC delta from block 0 :

– reading the first data bit

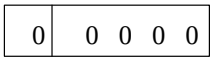
initial data stream (after header) →



– calculation of the address of the node (or leaf) of the HUFFMAN tree :

The address is formed by concatenating :

- a 4-bit register containing the 4 MSB's of the VLC word corresponding to the preceding node of the HUFFMAN tree (after shifting one bit right). For each new coefficient, the register is reset to 0.
- the last bit extracted from the data stream

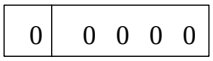


ADDRESS = 1st data bit (MSB) + contents of 4-bit register = 00H

– reading address 00H of the VLC DC table :

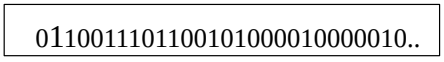
[@(00)] = 02H. This memory point corresponds to a node (in line with the conventions)

– right shift and load into 4-bit register :

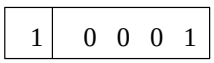


= 1st data bit + contents of the 4-bit register = 01H

– reading the 2nd data bit :
data stream →



– concatenation with the 4-bit register :



= 11H

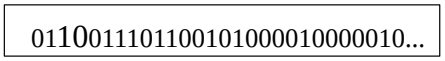
– reading address 11H

[@(11H)] = 05H. This memory point corresponds to a leaf (in line with the conventions)

– right shift :

ssss = 2 and the DC coefficient of block 0 is 2 bits long.

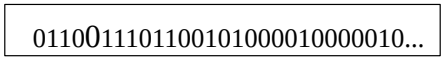
– extraction of the DC coefficient from the data stream :
data stream →



b2 - extraction of AC parameters

Extraction of AC delta from block 0 :

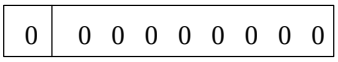
– reading the fifth data bit
initial data stream (after header) →



– calculation of the address of the node (or leaf) of the HUFFMAN tree :

The address is formed by concatenating :

- an 8-bit register containing the 8 MSB's of the VLC word corresponding to the preceding node of the HUFFMAN tree (after shifting one bit right. The LSB (bit 9) is obtained by address decoding). For each new coefficient, the 8-bit register is reset to 0.
- the last bit extracted from the data stream

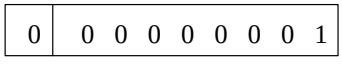


ADDRESS = 1st data bit (MSB) + contents of 8-bit register = 00H

– reading address 00H of the VLC AC table :

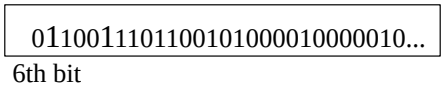
[@(00)] = 02H. This memory point corresponds to a node (in line with the conventions)

– right shift and load into 8-bit register :

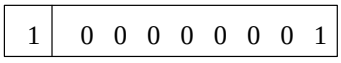


= 1st data bit + contents of the 8-bit register = 01H

– reading the 6th data bit :
data stream →



– concatenation with the 8-bit register :



= 101H

– reading address 101H

[@(101H)] = 03H. This memory point corresponds to a leaf (in line with the conventions)

– right shift :

nnnnssss = 00000001

- nnnn = 0. The number of zero coefficients = 0.
- ssss = 1. The first AC coefficient of block 0 is 1 bit long.

– extraction of the AC coefficient from the data stream : data stream →

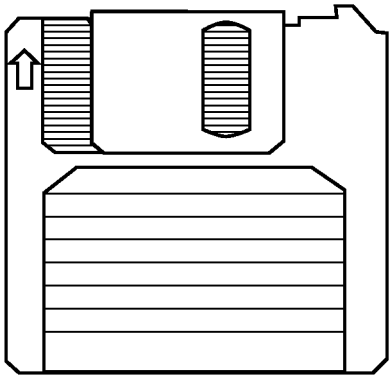
01**1**0011101100101000010000010..

AC_1_bloc_0 = 1

Note : nnnnssss = 00H corresponds to detection of an end-of-block (EOB) character. nnnnssss = 0FH corresponds to detection of an escape character (15 zero coefficients). In this case one has to set the following 15 coefficients to 0, and then resume the decoding sequence by re-initialising the 8-bit register to 0.

Appendix 4

Example of Program for Creation of VLC Table



```
/*   CONSTRUCTION OF VLC TABLES   */
/*      DC TABLES       */
void vlctable_dc (BITS, VALUE, HUFCODE, SIZE)
int             BITS[17] ;
int             VALUES[16] ;
int             HUFCODE[16] ;
{
int             PNTR ;
int             I,J ;
int             SIZO[16] ;
int             SI ;
int                     WORD ;
unsigned int CODE ;
for (I=0 < 16 ; I++)
                {SIZO[I] = 0 ;}
PNTR = 0 ;
for (I=1 ; I < 17 ; I++)
        for (J=1 ; J<=BITS[I] ; J++)
```

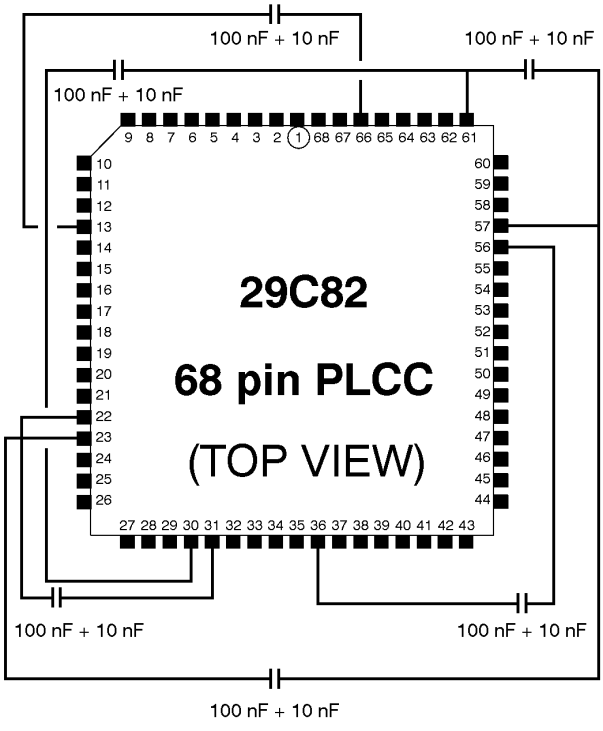
```
                {SIZO[PNTR] = I ; PNTR ++ ;}
for (I=0 ; I < 16 ; I++)
{HUFCODE[I] = 0 ; SIZE[I] = 0 ; }
WORD = 0 ;
CODE = 0 ;
SI = SIZO[0] ;
HUFCODE[VALUES[WORD]] = CODE ;
SIZE[VALUES[WORD]] = SI ;
do
{
                do
                {
                HUFCODE[VALUES[WORD]] = CODE ;
                SIZE[VALUES[WORD]] = SI ;
                CODE++ ;
                WORD++ ;
                }
                while (SIZO[WORD] == SI) ;
                if (SIZO[WORD] ! = 0)
                do
                {CODE = (CODE << 1) ; SI ++ ;}while (SIZO[WORD] ! = SI) ; else
                {break ;}
}
while (SIZO[WORD] == SI) ;
return ;
}
/*      DC TABLES */
void vlctables_ac (BITS, VALUES, HUFCODE, SIZE)
int             BITS[17] ;
int             VALUES[256] ;
int             HUFCODE[256] ;
int             SIZE[256] ;
{
int             PNTR ;
int             I,J ;
int             SIZO[256] ;
int             SI ;
int             WORD ;
unsigned int CODE ;
for (I=0 ; I < 256 ; I++)
        {SIZO[I] = 0 ;}
PNTR = 0 ;
for (I=1 ; I<17 ; I++)
        for (J=1 ; J<=BITS[I] ; J++)
        {SIZO[PNTR] = I ; PNTR ++ ;}
for (I=0 ; I < 256 ; I++)
        {HUFCODE[I] = SIZE[I] = 0 ;}
WORD = 0 ;
CODE = 0 ;
SI = SIZO[0] ;
HUFCODE[VALUES[WORD]] = CODE ;
SIZE[VALUES[WORD]] = SI ;
do
{
```

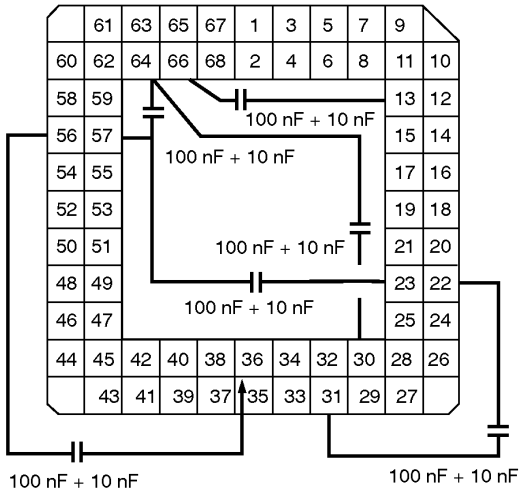
```
                do
                {
                HUFCODE[VALUES[WORD]] = CODE ;
                SIZE[VALUES[WORD]] = SI ;
                CODE++ ;
                WORD++ ;
                }
                while (SIZO[WORD] == SI) ;
if              (SIZO[WORD] ! = 0)
                        do
                        {CODE = (CODE << 1) ; SI ++ ; }while (SIZO[WORD] ! = SI) ; else
                        {break ;}
}
while (SIZO[WORD] == SI) ;
return ;
}
/*      CONSTRUCTION OF DECODING TREE                   */
/*      CONSTRUCTION OF DC TREE                 */
void makecodar_dc (HUFCODE, SIZE, PARR)
int             HUFCODE[16] ;
int             SIZE[16] ;
int             PARR[32] ;
{
int             INDEX ;
int             I ;
int             NEXT ;
int             CODENO ;
int             CODESIZE ;
int             CODELO ;
for (I=0 ;I < 32 ; I++)
                {PARR[I] = 0 ;}
NEXT = 0 ;
for (CODENO = 0 ; CODENO < 16 ; CODENO++)
                {
                if (SIZE[CODENO] ! = 0)
                        {
                                INDEX = 0 ;
                                CODESIZE = SIZE[CODENO] ;
                CODELO = HUFCODE[CODENO] ;
                CODELO = (CODELO << (16 - CODESIZE)) ;
                for (I=1; I < CODESIZE ; I++)
                                {
                                if (CODELO < 0)  INDEX += 16 ;
                                if (PARR[INDEX] != 0) INDEX = PARR[INDEX] ; else
                                                {       {
                                        NEXT ++ ;
                                        PARR[INDEX] = NEXT ;
                                        INDEX = NEXT ;
                                }
                                        CODELO = (CODELO << 1) ;
                        }
                if (CODELO < 0) INDEX += 16 ;
                PARR[INDEX] |= (CODENO << 8) ;
                        }
                }
```

```
return ;
}
/*              CONSTRUCTION OF AC TREE          */
void makecodar_ac (HUFCODE, SIZE, PARR)
int             HUFCODE[256] ;
int             SIZE[256] ;
int             PARR[512] ;
{
int             INDEX ;
int             I ;
int             NEXT ;
int             CODENO ;
int             CODESIZE ;
int             CODELO ;
for (I=0 ; I < 512 ; I++)
                {PARR[I] = 0 ;}
NEXT = 0 ;
for (CODENO = 0 ; CODENO < 256 ; CODENO++)
                {
                if (SIZE[CODENO] != 0)
                        {
                                INDEX = 0 ;
                                CODESIZE = SIZE[CODENO] ;
                                CODELO = HUFCODE[CODENO] ;
                                CODELO = (CODELO << (16 - CODESIZE)) ;
                                for (I=1 ; I < CODESIZE ; I++)
                                        {
                                        if (CODELO < 0)   INDEX += 256 ;
                                        if (PARR[INDEX] != 0) INDEX = PARR[INDEX] ; else
                                                        {
                                                NEXT ++ ;
                                                PARR[INDEX] = NEXT ;
                                                INDEX = NEXT ;
                                                }
                                                        CODELO = (CODELO << 1) ;
                                                }
                if (CODELO < 0) INDEX += 256 ;
                PARR[INDEX] |= (CODENO << 8) ;
                        }
        }
return ;
}
```

Appendix 5

Decoupling and Layout Precautions





It is recommended that ceramic capacitors of the CMS type should be used.

In order to improve decoupling, and to compensate for voltage peaks of inductive origin, a second 10 nF capacitor can be placed in parallel with each 100 nF capacitor. This additional capacitor should be placed as close as possible to the package.

Layout precautions

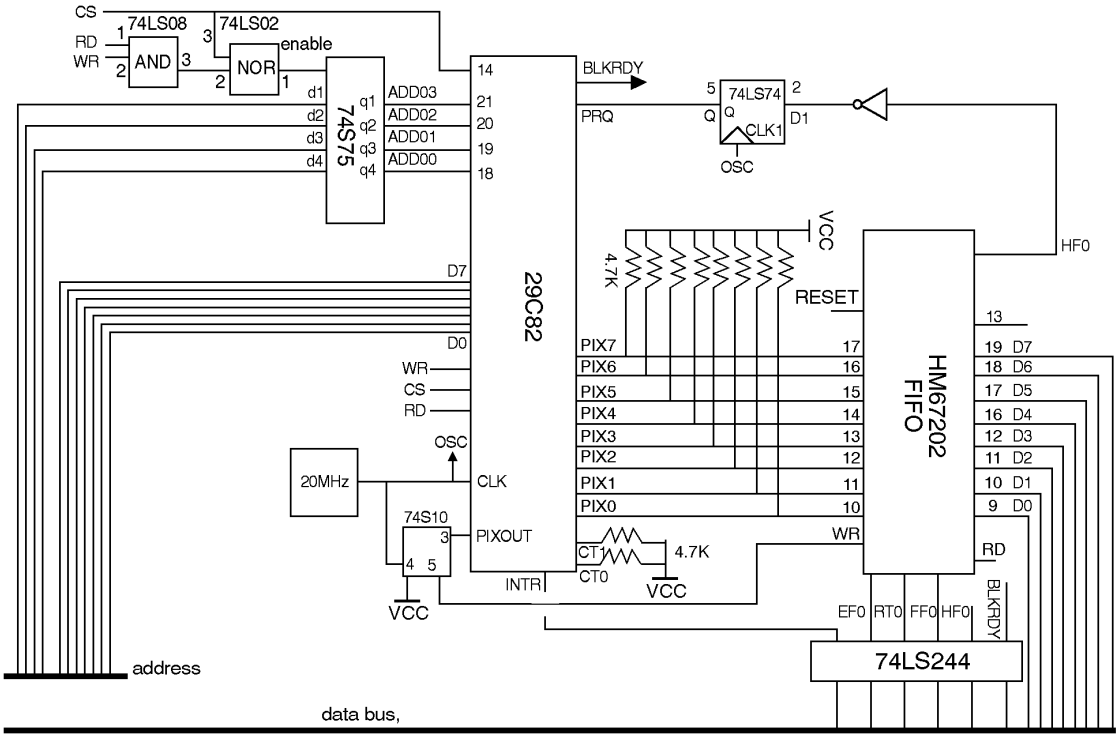
In order to reduce noise, it is recommended that a card with 4 levels of metalling be used, one level of which is reserved as the earth plane and a second for Vcc.

Appendix 6

Application

The use of an MHS HM67202 1k × 9 FIFO enables the frequency of accesses to the DATA bus to be reduced, thereby improving system flexibility.

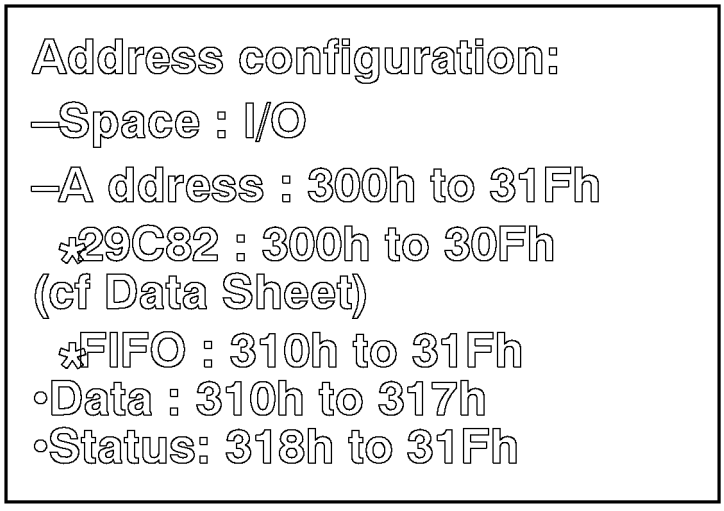
The latch placed on the ADD[3..0] bus is sampled on the rising edge of WR or RD (CS active). The HFO flag must be re-synchronised by the OSC signal (20 MHz clock) before it is injected into the PRQ input.



Appendix 7

29C82 PC Evaluation Board (8-bit)

– Schematic
– EPLD equation



C22V10

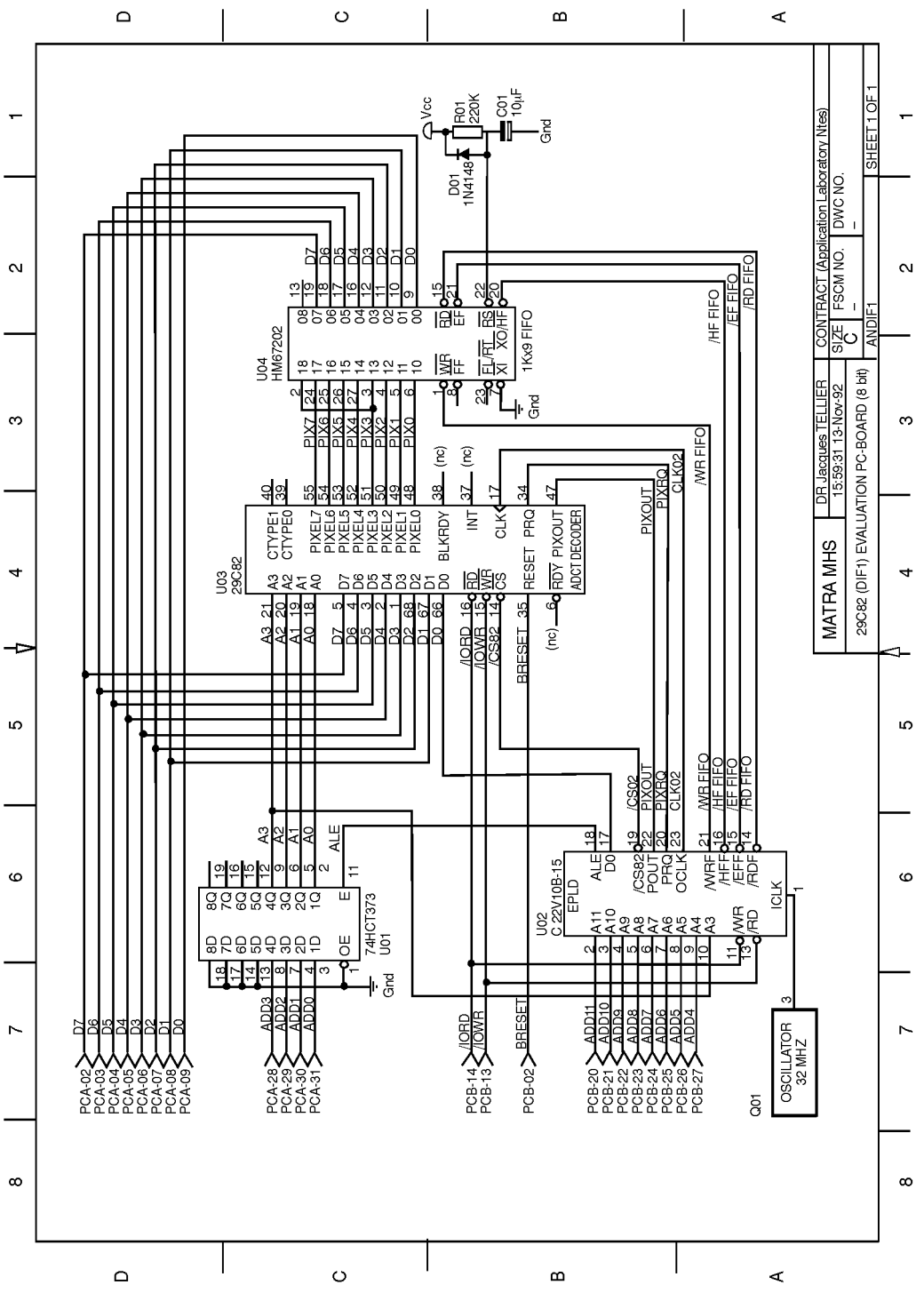
{29C82 Evaluation PC-Board}
{82EPC.CYP}
{On Board Logic & IO DECODING}
{Appli.Lab – J.T – 13.11.92}

CONFIGURE

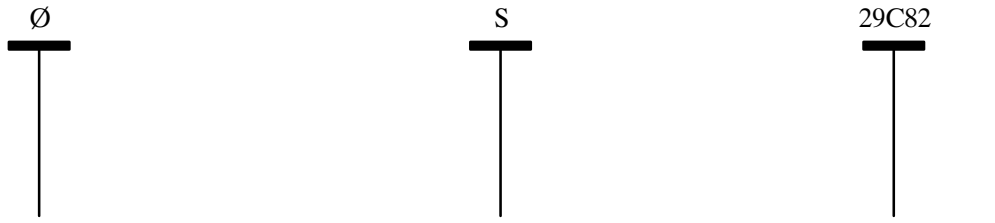
```
ICLK    (node = 1),   A11   (node = 2),   A10   (node = 3),
A9      (node = 4),   A8    (node = 5),   A7    (node = 6),
A3      (node = 10),  /WR   (node = 11),  /RD   (node = 13),
/RDF    (node = 14, NOREG),
/EFF    (node = 15, NOREG), {Input}
/HFF    (node = 16, NOREG), {Input}
D0      (node = 17, NOREG, NINV),
ALE     (node = 18, NOREG, NINV),
/C82    (node = 19, NOREG),
PRQ     (node = 20, NINV),
/WRF    (node = 21, NOREG),
POUT    (node = 22, NOREG), {Input}
OCLK    (node = 23, NOREG, NINV),
```

EQUATIONS

```
RDF =   {Read Control for FIFO Device. @IO = 310h to 317h.}
        <oe>
        <som> RD*/A11*/A10*A9*A8*/A7*/A6*/A5*A4*/A3 ;
D0 =    {reading Empty Flag from FIFO. @IO = 318h to 31Fh.}
        <oe> RD*/A11*/A10*A9*A8*/A7*/A6*/A5*A4*A3
        <sum> /EFF ;
C82 =   {Chip Select Control for 29C82 Device. @IO = 300h to 30Fh.}
        <oe>
        <sum> /A11*/A10*A9*A8*/A7*/A6*/A5*/A4 ;
ALE =   {Address Latch Enable for Latch Device type 74LS373.}
        <oe>
        <sum> /A11*/A10*A9*A8*/A7*/A6*/A5*RD
        <sum> /A11*/A10*A9*A8*/A7*/A6*/A5*WR ;
WRF =   {Write Control for FIFO Device.}
        <oe>
        <sum> /ICLK*POUT ;
PRQ =   {Pixel request to 29C82 synchronized by Clock.}
        {Available only if Half Full Flag from FIFO is not Present.}
        <oe>
        <sum> /HFF ;
OCLK =  {Clock for 29C82 synchronous with /WRF.}
        <oe>
        <sum> ICLK ;
```


MATRA MHS
DR Jacques TELLIER
15:59:31 13-Nov-92
CONTRACT (Application Laboratory Ntes)
SIZE C
FSCM NO. –
DWC NO. –
SHEET 1 OF 1
29C82 (DIF1) EVALUATION PC-BOARD (8 bit)
ANDIF1
U03 29C82
U04 HM67202
1Kx9 FIFO
U02 C 22V10B-15 EPLD
U01 74HCT373
Q01 OSCILLATOR 32 MHZ
D01 1N4148
R01 220K
C01 10µF
Vcc
Gnd
/HF FIFO
/RD FIFO
/EF FIFO
/WR FIFO
PIXOUT
PIXRQ
CLK02
/CS82
ALE
/IORD
/IOWR
BRESET
PCA-02
PCA-03
PCA-04
PCA-05
PCA-06
PCA-07
PCA-08
PCA-09
PCA-28
PCA-29
PCA-30
PCA-31
PCB-14
PCB-13
PCB-02
PCB-20
PCB-21
PCB-22
PCB-23
PCB-24
PCB-25
PCB-26
PCB-27

Ordering Information



TEMPERATURE RANGE
Ø COMMERCIAL 0 TO 70 °C
I INDUSTRIAL −40 TO 85 °C

PACKAGE : S = PLCC

Part number
29C82