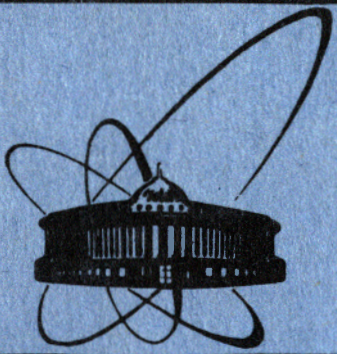


объединенный
институт
ядерных
исследований
дубна

E11-84-364

A.Leich, H.Leich, V.I.Prikhodko, M.Rudalics

A MICROPROGRAMMED BIT-SLICE ARITHMETICS PROCESSOR FOR FAST TRANSFORMATION OF GRAPHICAL ITEMS

Submitted to the Tenth Symposium on
Microprocessing and Microprogramming.
(Copenhagen, 1984)

1984

1. INTRODUCTION

In order to display graphics information on some graphics output device user data (usually given in world coordinates) are to be transformed to device coordinates. This paper describes the arithmetics tools of an Intelligent Graphics Terminal (IGT)/1/ capable of handling functions defined in the standard proposal "Graphical Kernel System"/2/ for creating a display list to be interpreted by a graphics processor (GP) which controls a vector display. The microprogrammed bit-slice arithmetics processor (AP) is a part of the arithmetics module (AM) of the IGT intended for calculations of standard mathematical functions and graphics transformations directly on microprogram level. It transforms normalized device coordinates (NDC) and attributes which are given in 32-bit floating point format to device coordinates (DC) and simple attribute information (16-bit fixed point format) handled by the GP.

2. ARITHMETICS MODULES STRUCTURE AND FUNCTIONS

The structure of the complete arithmetics module is represented in fig.1. It consists of a microcomputer based on a I8080 CPU which incorporates a 2K byte RAM and a 12K byte PROM, an interrupt system and an interface to the system bus of the IGT's multi-microprocessor system (MMPS)/1/. To unload the microprocessor when performing graphics transformations the microprogrammed bit-slice arithmetics processor (AP) is attached to the internal bus of the microcomputer. To compensate the various operation speed of CPU and AP graphical data as well as transformation parameters are routed from the IGT's common memory to the AP via a special buffer memory (BM) of 1K byte capacity. The data transfer is performed by means of a DMA-unit under the control of the CPU. Communication between CPU and AP is established via a 16-byte control-register (CR) file - a special dual port memory providing simultaneously access for the CPU and the AP. To activate a macrofunction in the AP the CPU writes the code and possible parameters of the desired function to predefined registers of the CR-file. After completing the execution of the task the AP puts a status information into the CR-file and fetches the code of the next operation to be performed. Transformed data (coordinates and

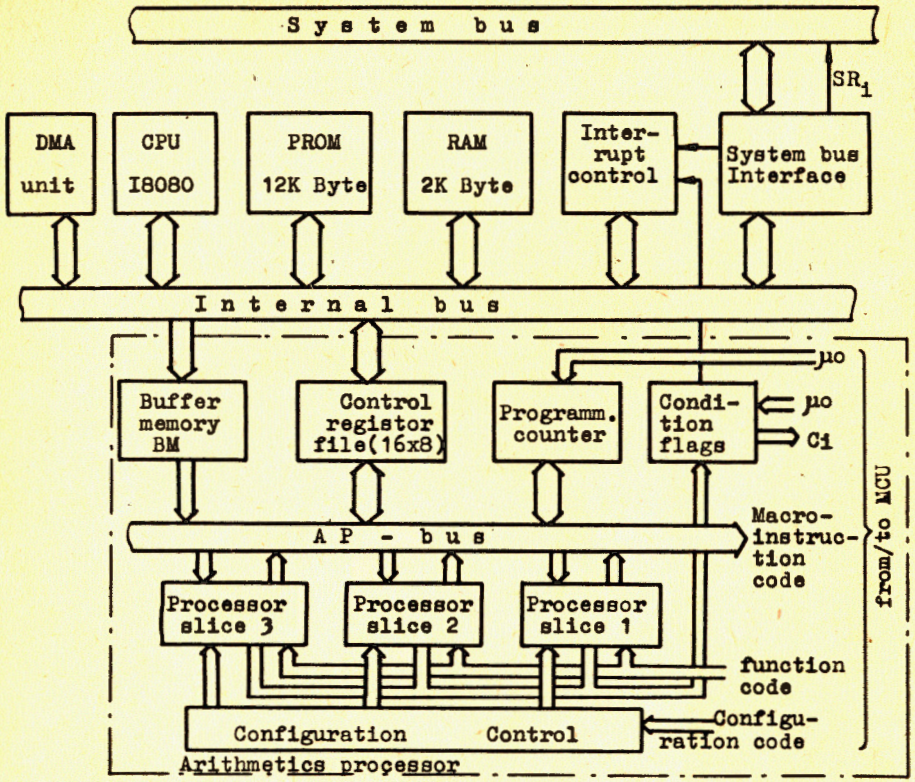


Fig.1. The architecture of the AM.

attributes) are passed back to the common memory via the CR-file, one register of which serves as an output buffer. Everytime one byte is written from the processor to that register a request to a separate channel of the DMA-unit is generated causing a data transfer from the CR-file to the common memory without CPU intervention.

The IGT performs functions of a GKS workstation (GKS-terms used in this paper are printed by capital letters). To achieve highly autonomous performing of all transform operations the AM handles all GKS functions which parameters influence the transformation of graphics data. Floating point values of these parameters are held in the BM.

The functions of the AM's CPU are the following:

- to initialize and to update the transformation parameters and output attributes;

- to control the generation of the display list on the base of GKS functions as WORKSTATION CONTROL FUNCTIONS, OUTPUT FUNCTIONS, TRANSFORMATION FUNCTIONS, SEGMENT FUNCTIONS, INPUT FUNCTIONS;
- to control the display list regeneration on the base of a data structure residing in the IGT's common memory /3/;
- to activate the AP to perform arithmetics operations.

During display list regeneration a part of the buffer memory is used also as a stack for transformation matrices.

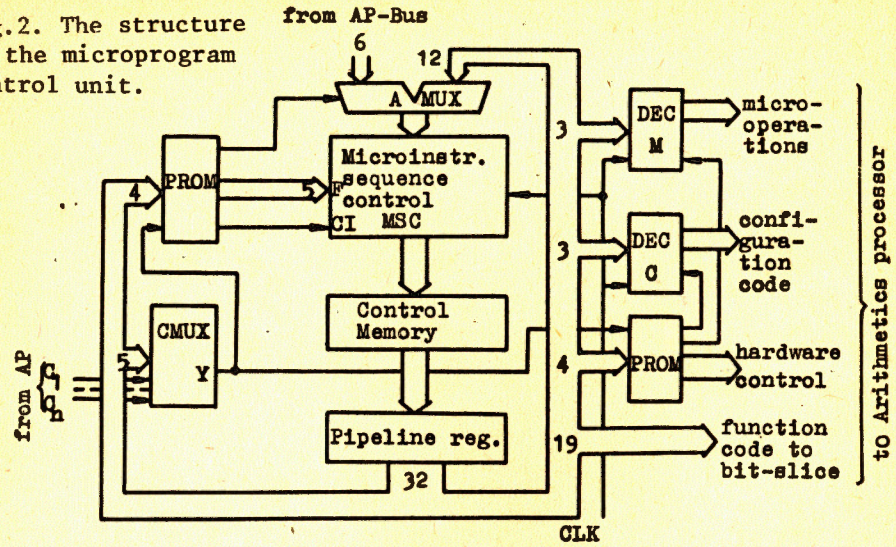
3. AP-HARDWARE

The execution unit of the AP incorporates three 8-bit-wide slices built up with Am2901 processor elements. Using a special field of the microinstruction word which activates the desired slices by means of the configuration control hardware the user (i.e., the firmware programmer) has the opportunity, to perform operations in either of the slices or in combinations of them. Thus an effective mechanism is provided to handle varibus data formats. Integer numbers are processed on slices 1 and 2. For floating point data a 32-bit format (pdp-format) is used. The mantissa (24 bits) is processed in parallel on all three slices while the exponent is calculated on slice 1. Data transfer operations to or from the AP are performed via the 8-bit AP-bus. Prior to receive information from the BM the base address of the desired parameter block or block of coordinates is to be loaded into an address register by means of special microinstructions. This address register is autoincremented everytime one byte is fetched from the BM.

The operation of the arithmetics processor is controlled by a microprogram control unit (MCU). It is built around three Am2911-sequencer elements (fig.2). Several microinstruction formats are used to reduce the microinstruction word length and to provide comfort of microprogramming similar to that which is provided by assembler languages. The first four bits determine how to interpret the other fields of the microinstruction word (fig.3). A list of various microinstruction types is presented in fig.4. The sequence of microinstructions may be altered at execution time depending on the actual state of certain conditions (fig.5).

Implementing the MCU special attention was given to the fast execution of time-consuming basic operations like normalization/denormalization, multiply and divide. A special counter is provided for performing iterative microinstructions like DO WHILE (DOW) (fig.4) and LOOP ON CONDITION (LOC). It may be loaded from the MCU or from a processor slice. The microinstruc-

Fig.2. The structure of the microprogram control unit.



tion DOW may be combined with any processor instruction and causes the repeated execution of the specified action until the selected condition changes from true to false. Everytime a DOW or LOC instruction is executed the counter is automatically incremented. The actual state of counter is available to the processor for further calculations, for instance to correct the exponent after normalization-shifting. Using the counter overflow condition and properly loading the counter a predefinednumber of cycles may be executed (to realize multiply, divide as well as other functions). Special hardware supports the implementation of multiply and divide algorithms.

As a result of all mentioned facilities the following PL/M-statements may be executed in one clock-cycle:

```
DO WHILE Ci = { TRUE
              { FALSE };

   IF Cj = { TRUE
           { FALSE }

      THEN operation 1;
      ELSE operation 2;
END;
```

When a CALL instruction is executed, the actual count value is stacked. A RET operation restores the previous value.

The LSU (LOOP SET UP) instruction stores the entry point of a microprogram loop in a special register of the MCU. The LOC instruction examines the specified condition, and if it is true a jump to the loop entry point is performed.

MICROINSTRUCTION FORMATS

FORMAT: BITPOSITION:

	31-28	27 --- 19	18--14	13-12	11-9	8 7 6 5 4 --- 0
PROC	OPC	S/F/D	CNR	SI,CI	SC	RAMB / RAMA/ CR-NR.
SEQC	OPC	0	CNR	0	JMPADDR	
FETCH	OPC	0	0	0	OFFSET	CR-NR.
AUX	OPC	0		CC	CONST	

OPC: OPERATION CODE; S/F/D: SOURCE-FUNCTION-DESTINATION CODE FOR THE BIT-SLICE; CNR: NUMBER OF THE CONDITION TO BE EXAMINED; SI/CI: SHIFT-/CARRY-INPUT TO THE BIT-SLICE; SC: SLICE-CONTROL; RAMB, RAMA: B-/A-ADDRESS TO THE DUAL-PORT REGISTER FILE OF THE BIT-SLICE; CC: CONFIGURATION CONTROL JMPADDR: JUMP-ADDRESS; OFFSET: 6 HIGH-ORDER-BITS OF THE DESIRED JUMP ADDRESS (6 LOW ORDER BITS ARE TAKEN FROM THE SPECIFIED CR); CR-NR.: ADDRESS TO THE CONTROL REGISTER FILE
CONST: ANY CONSTANT VALUE (ONLY LAST 8 BIT USED)

Fig. 3

SEQUENCE-CONTROL INSTRUCTIONS (SEQC-FORMAT):

(JUMP, IF COND = TRUE)

JOC: JUMP ON CONDITION
JMP: JUMP ABSOLUTE
COC: CALL ON CONDITION
CALL: CALL SUBROUTINE
ROC: RETURN ON CONDITION
RET: RETURN FROM SUBROUTINE
JEXT: JUMP TO ADDRESS FROM EXTERNAL SOURCE (FETCH-FORMAT)

BIT-SLICE PROCESSSOR INSTRUCTIONS (PROC-FORMAT):

DO: EXECUTE A PROCESSOR INSTRUCTION (ADD, SUB, SHIFT, INCR, DECR, AND, OR ...) AND CONTINUE
DOW: EXECUTE A PROCESSOR INSTRUCTION WHILE THE SELECTED CONDITION TRUE
LSU: EXECUTE & STORE ENTRY POINT OF A MICROPROGRAM LOOP
LOC: GO TO ENTRY POINT OF A MICROPROGRAM LOOP, IF THE SPECIFIED CONDITION IS TRUE

AUXILARY INSTRUCTIONS (AUX- OR PROC-FORMATS)

LABM : LOAD BUFFER-MEMORY ADDRESS FROM MCU
LABMP: LOAD BUFFER-MEMORY ADDRESS FROM PROCESSOR-SLICE
WRCTR: LOAD COUNTER FROM PROCESSOR-SLICE
LDCTR: LOAD COUNTER FROM MCU
RDMEM: LOAD PROCESSOR-SLICE FROM BUFFER-MEMORY,
RDCTR: LOAD PROCESSOR-SLICE FROM COUNTER
RDCR: READ FROM CONTROL REGISTER FILE
WRCR: WRITE TO CONTROL REGISTER FILE

Fig. 4

Table

Operation	Execution time in µs			
	AP	SBC 310	AM 9511	I8087
FADD	11	75	185	24
FSUB	11	75	185	24
FMUL	9	100	84	25
FDIV	9	110	92	45
FSQRT	75	205	435	37
FSIN	300	-	2400	-
FCOS	300	-	2400	-

MCU-CONDITIONS

(ALL CONDITIONS ALSO INVERTED AVAILABLE)

CNR	MNEMONIC	CONDITION
0	FALSE	FALSE-CONDITION
1	SIGN1	SIGN OF PROCESSOR-SLICE 1
2	SIGN2	SIGN OF PROCESSOR-SLICE 2
3	SIGN3	SIGN OF PROCESSOR-SLICE 3
4	CARRY3	CARRY-OUT FROM SLICE 3
5	OFL1	OVERFLOW FROM PROCESSOR-SLICE 1
6	OFL2	- " - 2
7	ZERO1	ZERO OF PROCESSOR-SLICE 1
8	ZERO2	- " - 2
9	ZERO3	- " - 3
10	NOZERO21	NOZERO - " - 1&2
11	NOZERO	NOZERO (ALL SLICES)
12	RAM0	RAM-SHIFTER BIT 0
13	RAM23	RAM-SHIFTER BIT 23
14	CTROFL	COUNTER-OVERFLOW
15	DMABUSY	DMA - BUSY

Fig. 5

Based on the operations ADD, SUB, MUL and DIV higher mathematical functions as well as complex graphics transformations are calculated using subroutine technique. In the table some speed characteristics of the AP in comparison with other arithmetics processors are provided (operations with 32-bit floating point data).

4. FUNCTIONS IMPLEMENTED BY MICROCODE

The arithmetics processor handles the following output primitives:

POLYLINE - a set of connected straight lines given by the number of connected points and a sequence of NDC coordinate pairs;

POLYMARKER - symbols of one type centred at a sequence of positions given by the number of points and a sequence of NDC coordinate pairs;

TEXT - a character string at a defined position given by a sequence of character symbol codes and an NDC coordinate pair.

The output primitives FILL AREA and CELL ARRAY are simulated by POLYLINE.

The following attributes are handled by the AP:

for POLYMARKER:

- MARKER SIZE SCALE FACTOR (a factor by which the nominal marker size is to be multiplied)

for TEXT (to realize TEXT PRECISION CHAR[3]):

- CHARACTER HEIGHT;
- CHARACTER UP VECTOR (gives the up direction of a character);
- CHARACTER EXPANSION FACTOR (specifies the deviation of the width to height ratio of the character from the ratio indicated by the font designer);
- CHARACTER SPACING (a fraction of the font nominal character size; it specifies how much additional space is to be entered between two adjacent character bodies);
- TEXT PATH (specifies the writing direction of the text string - RIGHT, LEFT, UP, DOWN);
- TEXT ALIGNMENT (controls the positioning of the TEXT EXTENT RECTANGLE - an imaginary box enclosing the text string - in relation to the text position; it consists of a horizontal component and a vertical component).

The transformations are performed depending on the current TRANSFORMATION MATRIX and are given by the following matrix multiplication:

$$\begin{bmatrix} X1 \\ Y1 \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

where A and E define the scaling factors
A, B, D and E define some rotation
C and F define the displacement values

Clipping is performed in the device coordinate space before data are converted to fixed point format. POLYLINE output

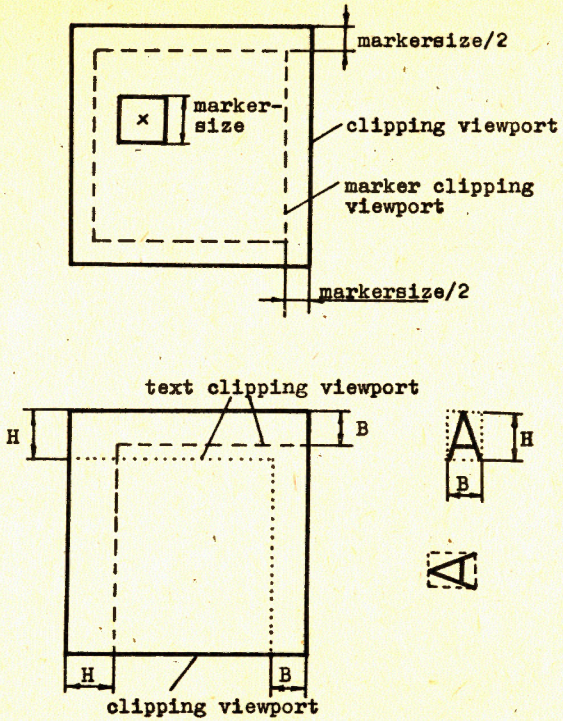


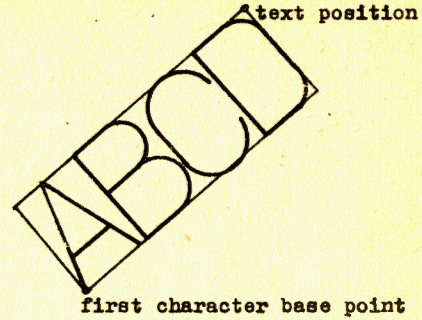
Fig.6. Marker and text clipping viewport.

primitives are clipped following the algorithm described in /4/. For clipping of markers and text symbols the clipping viewport is changed to a marker clipping viewport or a text clipping viewport (fig.6). This allows to eliminate the symbol size dependency from the clipping procedure.

To provide functional complexity on microprogram level a set of floating point valued is held in the BM of the bit-slice processor, for instance:

- CLIPPING RECTANGLE	(two NDC coordinate pairs)
- WORKSTATION WINDOW	(two NDC coordinate pairs)
- WORKSTATION VIEWPORT	(two NDC coordinate pairs)
- clipping window	(two NDC coordinate pairs)
- clipping viewport	(two NDC coordinate pairs)
- marker clipping viewport	(two NDC coordinate pairs)
- text clipping viewport	(two NDC coordinate pairs)
- WORKSTATION TRANSFORMATION MATRIX	(four NDC values)
- SEGMENT TRANSFORMATION MATRIX	(six NDC values)
- instance transformation matrix /3/	(six NDC values)
- active transformation matrix	(six NDC values)
- CHARACTER SPACING	(one floating point value)

Fig.7. Relationship between text position and first character base point for TEXT PATH=RIGHT and TEXT ALIGNMENT = RIGHT, TOP.



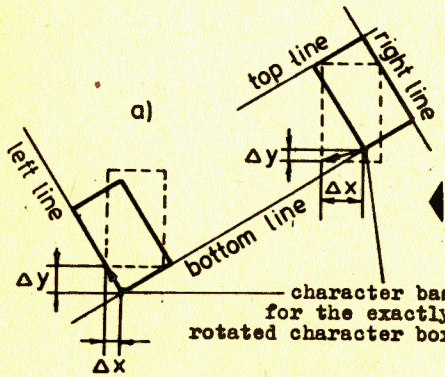


Fig.8. Displacement (Δx, Δy) of the character base point dependent on the TEXT ALIGNMENT. a) TEXT ALIGNMENT = LEFT, BOTTOM, b) TEXT ALIGNMENT = = RIGHT, TOP.

- character vectors	(four floating point values)
instead of CHARACTER UP VECTOR and CHARACTER HEIGHT	
- CHARACTER EXPANSION FACTOR	(one floating point value)
- TEXT POSITION	(one NDC coordinate pair)
- MARKER SIZE SCALE FACTOR	(one floating point value).

The following functions are implemented by microcode:
- calculation of the WORKSTATION TRANSFORMATION MATRIX;
- calculation of a TRANSFORMATION MATRIX inverse to the WORKSTATION TRANSFORMATION MATRIX;
- calculation of the TRANSFORMATION MATRIX active for the current output primitive (accumulation of SEGMENT TRANSFORMATION MATRIX, INSERT TRANSFORMATION MATRIX and WORKSTATION TRANSFORMATION MATRIX may be performed);
- calculation of clipping boundaries on the base of CLIPPING RECTANGLE and WORKSTATION WINDOW;
- transformation of the CHARACTER HEIGHT of text symbols and of the markersize given by the MARKER SIZE SCALE FACTOR to a fixed point value in the range 0...15 which may be directly interpreted by the graphics processor;
- transformation of the TEXT OUTPUT ATTRIBUTES TEXT PATH, CHARACTER HEIGHT, CHARACTER UP VECTOR, CHARACTER EXPANSION FACTOR and CHARACTER SPACING to the values handled by the GP:

character displacement - displacement between the base points of two adjacent character bodies

text direction - symbols may appeare on the screen rotated by 0, 90, 180 and 270 degrees;

- calculation of the basepoint coordinates of the first character of a character string according to the TEXT ALIGNMENT and TEXT PATH attributes (fig.7);

- defining a displacement of the character basepoint in such a way that symbols situated along an inclined baseline and arranged in one of the four text directions woun't overlap the alignment boundaries (fig.8);

- functions performing transformation and clipping of the output primitives POLYLINE, POLYMARKER and TEXT.

When handling output primitives like POLYLINE and POLYMARKER the CPU splits the set of NDC coordinates into portions and writes them to one of two buffer areas in the BM. While writing data to one buffer data from the other buffer are transformed by the AP without CPU intervention.

5. CONCLUSIONS

The microprogrammed bit-slice processor performs complex transformations like displacement, scaling, rotation, window/viewport transformation and clipping on microprogram level based on subroutines for standard mathematical functions. It provides an effective mechanism to handle various data formats and incorporates facilities for fast execution of time-consuming operations. Functional complexity on microprogram level is achieved by providing the AP direct access to transformation parameters and operands.

REFERENCES

1. Leich H., Levchanovsky F.V., Prikhodko V.I. A Multi-Microprocessor Based Intelligent Graphics Terminal. Microprocessing and Microprogramming 12, 1983, 3-4, pp. 175-180.
2. Draft International Standard ISO/DIS 7942, Information Processing Graphical Kernel System (GKS), Functional Description, Version 7.2, NI-5.9/1-83, Nov. 1982.
3. Rudalics M. JINR, E11-83-393, Dubna, 1983; Also: Proc. of the Eurographics'83, North-Holland Pub., 1983, pp. 383-392.
4. Newman W.M., Sproull R.F. Principles of Interactive Computer Graphics. McGraw-Hill, 1973.

Received by Publishing Department
on May 24, 1984.

SUBJECT CATEGORIES OF THE JINR PUBLICATIONS

Index	Subject
1.	High energy experimental physics
2.	High energy theoretical physics
3.	Low energy experimental physics
4.	Low energy theoretical physics
5.	Mathematics
6.	Nuclear spectroscopy and radiochemistry
7.	Heavy ion physics
8.	Cryogenics
9.	Accelerators
10.	Automatization of data processing
11.	Computing mathematics and technique
12.	Chemistry
13.	Experimental techniques and methods
14.	Solid state physics. Liquids
15.	Experimental physics of nuclear reactions at low energies
16.	Health physics. Shieldings
17.	Theory of condenced matter
18.	Applied researches
19.	Biophysics

Лайх А. и др. E11-84-364

Микропрограммируемый арифметический процессор для быстрого преобразования графических объектов

Мультимикропроцессорный интеллектуальный графический терминал /ИГТ/, разрабатываемый в ОИЯИ, автономно выполняет сложные графические преобразования. Операции типа смещение, масштабирование, вращение, преобразования для выделения окна данных и окна экрана, заданные матрицей преобразования в соответствии со стандартом Graphical Kernel System, осуществляются в специальном арифметическом модуле ИГТ, который обеспечивает вычислительные ресурсы для выполнения указанных преобразований. Модуль состоит из универсального микропроцессора, который может анализировать и обрабатывать сложные структуры данных, и быстрого микропрограммируемого арифметического процессора для вычисления стандартных математических функций и выполнения графических преобразований непосредственно на микропрограммном уровне. Арифметический процессор построен на элементах серии Am 2900 /КР1804/.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Препринт Объединенного института ядерных исследований. Дубна 1984

Leich A. et al. E11-84-364

A Microprogrammed Bit-Slice Arithmetics Processor for Fast Transformation of Graphical Items

The multi-microprocessor based intelligent graphics terminal (IGT) designed at the JINR at Dubna autonomously performs complex graphics transformations. Operations like clipping as well as displacement, scaling, rotation and window/viewport transformation defined by a transformation matrix and also attributes handling as given by standard "Graphical Kernel System" are implemented on a special arithmetics module of the IGT, which provides the computing resources for performing mentioned transformations. The module consists of an universal microprocessor, which is capable of analyzing and handling complex data structures, and a high-speed microprogrammed arithmetics processor for calculations of standard mathematical functions and performing graphics transformations directly on the microprogram level. The arithmetics processor is built up with elements of the Am2900 (KR1804)-bit-slice family.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Preprint of the Joint Institute for Nuclear Research. Dubna 1984