COMMISSARIAT A L'ENERGIE ATOMIQUE

CENTRE D'ETUDES NUCLEAIRES DE SACLAY
Service de Documentation
F91191 GIF SUR YVETTE CEDEX

CEA-CONF-- 9081

L4

CEA-DPh-N-S--2443B

THE GRAPHICS SOFTWARE OF THE SACLAY LINEAR ACCELERATOR CONTROL SYSTEM

Gournay, J.F. CEA CEN Saclay, 91-Gif-sur-Yvette (France). Service de l'Acceierateur Lineaire

Communication présentée à : 7. Summer school on computing techniques in physics
Bechyne (Czechoslovakia)
9-18 Jun 1987

THE GRAPHICS SOFTWARE OF THE SACLAY LINEAR ACCELERATOR CONTROL SYSTEM

J.F. GOURNAY
Service de l'Accélérateur Linéaire
CEN - SACLAY - 91191 GIF SUR YVETTE - FRANCE

INTRODUCTION

The Control system of the Saclay Linear Accelerator [1] is based upon modern technology hardware. It consists of MC 68000 microprocessors housed either in 16/32 bit VME crates or in 8/16 bit Euromak crates. The tasks are distributed among these different subsystems in a functional and geographical manner. At the upper level 3 VME crates interconnected by parallel links are responsible for 3 specific group of tasks : data base management, consoles management and control/acquisition supervision. At the lower level, 12 Euromak crates linked together with a MIL-1553B network and located along the accelerator are responsible for equipment control.

The basic console consists of a color TV monitor, a touch panel, 2 knobs and a keyboard. The man-machine interface has been carefully designed to insure good flexibility, fast response and good quality displays. To fulfill these criteria, a graphics software package was written. It was influenced by many concepts of GKS [2] which was just an emerging standard in 1983 when this work started. The Unified Graphics System[3] that I used at SLAC in 1981/1982 has been also a good help especially for designing the interface between the graphics software and the application programs.

THE GRAPHIC SOFTWARE

Pictures are created in exactly the same manner for all the graphic devices supported by the system. The informations used to draw a picture are stored in an array called a graphic segment.

1 - Segments management.

Two subroutines are directly related to the management of these graphic segments :

- GKINI : This subroutine is used to clear and initiatize a segment. A single parameter is passed to it : an integer value in the range 1 to 20 which is the segment number.

- GKXMIT : This subroutine transmits a previously constructed graphic segment to a device driver. This driver translates the segment in device-dependent orders. Three parameters must be present in the calling sequence :

  - an option parameter with any of the following values
    ± 1 : draw a segment
    ± 2 : erase a segment that has previously been drawn
    ± 3 : clear the display device and then draw the segment.

If the parameter value is positive the segment is closed after having been transmitted to the driver. If the parameter value is negative the segment is always available. This feature is particularly useful for the management of dynamic displays : the segment is drawn with option -1 and later, just before transmitting a new picture, it is erased with option 2 without the overhead to reconstruct it.

- the segment number.
- the identifier of the device driver.

2 - Output Primitives

Three output primitives are used to add graphic material in a segment:

- GKTEXT : This subroutine draws a character string at a given position. To insure a good response time required with interactive and refreshed display lists, the characters are generated by the hardware. Six parameters are passed to this subroutine :

  - An option string which way contain any of the following items

  TX = <character horizontal size>, TY = <character vertical size>.
  TX and TY are relative to the default window and then are independant of any coordinates transformation.
  FX = <character horizontal size>, FY = <character vertical size>, relative to the current window.
  C = <character colour>, F = <background colour>,
  I = <tilting>, D = <orientation of the characters> ,
  J = <justification> (left,center,right).

  - The segment number.

  - X, Y : the coordinates to locate the string.

  - N : character string length.

  - TEXT : character string.

- GKVECT : This subroutine generates a set of connected vectors. Six parameters are passed to this subroutine :

  - An option string which may contain any of the following items

  TYP = <linestyle> (solid, bold, dotted...)
  C = <colour>

  - The segment number.

  - N : number of vectors.

  - X, Y : arrays containing the X, Y coordinates.

  - an array of blanking flags
    if flag (I) = 1 draw vector I
    else if flag (I) = 0 move to coordinates X(I),Y(I)
    else if flag (I) = -1 draw vector I to vector N.

- GKFILL : This subroutine is used to fill a rectangular area.
  Six parameters are passed to this subroutine :
  - An option string which may contain the coulour of the area (C=) and its interior style (TYP=).
  - The segment number.
  - The coordinates of the 2 opposite corners of the rectangle.

3 - Coordinate Systems and Transformations

Like most current graphics software three coordinate systems are defined

- the world coordinate ("WC" according to the GKS denomination) used by the application software.

- the normalized device coordinate ("NDC") an intermediate and device independent system. All the coordinates in the graphic segments are expressed in this system. Its range is $[0.,1000.[$ in the X and Y directions.

- the device coordinate ("DC") related to the specific hardware. The device drivers are responsible of the transformation between the NDC and DC systems.

The subroutine GKWDW is used to define a rectangular subset of the NDC space and map it into an arbitrary area of the WC system. This transformation is identified by an integer value in the range $[1,20]$.

The subroutine GKSELW selects a transformation previously defined by GKWDW on a given segment. If some elements of the graphic segment on which the transformation applies extend outside of the current window, they are clipped at the window boundaries. By default there is no transformation between the WC and NDC systems.

A third subroutine combines the transformation selection of GKSELW with the possibility to manage multiple overlapping windows.

4 - Control of the Graphic Devices

One subroutine available to the application programs is particular in the sense that it does not deal with the segments, but rather directly sends orders to the device driver. The possible options are : clear the display, selectively clear a colour memory plan, test or reset the graphic controller.

5 - Implementation

The graphics software is run in the Console VME station as a server task. The programs have access to it either through local messages (e.g. management of a touch-panel) or through remote messages (application programs located in a remote VME crate). The software is written in FORTRAN 77 with the exception of the device drivers which are written in MC68000 Assembly language. Device drivers are available for the Eltec GRAZ 3/4 VME modules, for the Data Sud CHROMA8 IPC VME module, for the Tektronix 4662 plotter and for the Visual 550 graphic display terminal.

All the informations concerning the graphic devices are stored in the database of the system (device driver name, resolutions, colour capabilities, address, segments). Each virtual device of the system (Main_Visual_1, Main_Visual_2, Alarm_Visual, Touch_Panel_1,...) is associated with an entry in the database. If necessary, some entries can be modified without the need of recompiling any piece of the software. Furthermore, a program can redirect its outputs only in changing the virtual device name (typically from Main_Visual_1 to Main_Visual_2 for an application program or from Touch_Panel_1 to Touch_Panel_2 for the touch panel handler).

CONCLUSION

After 2 years of experience in using this software, its major advantages can be quoted :

- dividing a picture into several segments and being able to individually draw or erase them, is very convenient for refreshed pictures. A rate of 10Hz of refreshment is easily reached in the case of rather simple pictures.

- working in real world coordinates relieves the high level application programs of cumbersome transformations.

- the small number of primitives and the extensive use of character strings for arguments are features which give very easy to read programs.

- redirecting outputs only in modifying a single parameter in the application programs gives a great flexibility.

REFERENCES

1) J.F. GOURNAY et al., - Proc. of the 1985 Particle Accelerator Conference Vancouver, Trans. Nucl. Sci. NS-32 n°5 p 2026

2) Graphical Kernel System (GKS) - Functional Description Draft International Standard ISO/OIS 7942

3) The Unified Graphics System for FORTRAN 77
Robert C. BEACH - SLAC Stanford, CAL 94305