EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH

CERN - PS DIVISION

22

CERN LIBRARIES, GENEVA

CERN/PS 93-50 (CO)

su 9348
G

CM-P00059544

AN INEXPENSIVE ANALOG FUNCTION GENERATOR IN VME STANDARD

W. Heinze, R. Maccaferri

International Conference on Accelerator and Large Experimental Physics Control
Systems
Berlin, Germany, October 18-22, 1993

Geneva, Switzerland
15/11/93

AN INEXPENSIVE ANALOG FUNCTION GENERATOR IN VME STANDARD

W. Heinze, R. Maccaferri, PS Division CERN, CH-1211 Geneva 23, Switzerland

Abstract

As part of the rejuvenation of the CERN accelerators control system, a simplified analog function generator ("SAFG" or "GFAS" as known at CERN in French) has been developed in VME standard, replacing the 15 years old model. Most of them will be used to control the current in magnets. In order to be usable for all CERN configurations, it is split in two parts. The part in the VME module generates the function in digital form, the instantaneous digital values are then transmitted in real-time over a serial link to a DAC module that generates the actual analog function. The DAC is very simple and exists in VME, G64 and single Europe standard. It can be easily implemented in any other standard thus allowing for adaptation to all existing configurations where a function generator is needed. The analog function is approximated by vectors; up to 512 vectors can be used per function and up to 32 functions can be stored at one time for each function generator; each vector is approximated by a staircase function with 5 μ s step duration or a multiple of it.

1. Introduction

At CERN, we find several function generators [1,2,3,4], based on two different methods: hardware multiplication [1,2], and CPU real time calculation [3,4]. The analysis of the actual CERN needs [5] shows that, in most applications, we do not need an external clock if the internal time resolution is precise enough ($\sim 5\mu$ s). However, the number of different functions should be >16 and number of vectors per function >128 , to satisfy the various requirements of our accelerators. The function generator must be electrically insulated from the user application, and must be in VME standard to be integrated into the new control system. Because of the high number of function generators needed, they should be inexpensive. Therefore, the GFAS uses CPU logic to calculate and produce the function in real-time. With this function generation method, the number of vectors and functions is only limited by the internal memory size, and the step size by the CPU speed. To minimize the cost further, two function generator channels are implemented in a single width VME module.

2. Function definition

Each vector is defined in a Cartesian notation $\Delta V, \Delta t$. ΔV is a 16 bit signed integer and the Δt value is composed of two 16 bit unsigned integers, NS = number of steps, and SZ = step size. SZ is the multiplicity of the basis step size, thus the duration of a vector is $\Delta t = NS \times SZ \times 5\mu$ s. The end of the function is indicated by NS = SZ = 0. Finally, the function origin (i. e. the value at

which the function starts) is defined by V0 which is a 16 bit signed integer (fig. 1). One function can be composed of up to 512 vectors, up to 32 different functions can be stored per GFAS channel.

Fig.1 Vector and function definition

3. Operation of the GFAS

Function writing:

The functions are written as a sequence of triplets of numbers defining the successive vectors to be produced. The sequence is terminated by writing a zeroed triplet. One or more functions can be written via the VME bus to the module at any time but they will be transferred to the internal CPU memory only when it is not generating a function. The status indicating the internal CPU activities can be read via the VMEbus.

Function start:

Once a valid function is selected (by writing the number of the function into a mailbox address), the GFAS sends the origin value V0 to the DAC and waits for an internal (software!) or external start pulse which starts the function generation. The function will proceed until the last defined vector has been produced. Then the origin value V0 is restored. Hereafter, a new cycle can be started with a new start pulse.

Function generation:

Once started, the CPU enters into a intelligently programmed loop to calculate the momentary values at the exact time intervals given by the step size. All possible CPU activities, e. g. taking the next vector, are programmed so that the step size is exactly maintained.

Function abort:

Function generation can be aborted at any time by a stop pulse which terminates the cycle and restores the origin value V0 immediately.

Function suspend:

Function generation can be suspended by an external intermediate stop, called "event stop": the actual value is kept until an intermediate start ("event start") occurs. The function will then continue normally until the end, see fig. 2. If in the suspended state an abort arises, then the function will stop completely as described above.

Function selection:

A new function selection (by writing the number of the function into the mailbox address) is only effective when the generation is stopped, but the behaviour will be different if the function generation is waiting for a normal start or for an event start. In the first case, the origin value V0 of the new function is applied immediately and the generator waits for a start. In the second case, the output value will reach the new origin value with a linear interpolation with a slope $\Delta V/\Delta t$ defined by a special parameter, and a subsequent event start will execute the new function from the beginning. The select command validates the various modes described below, see fig. 2.

4. Other operating modes

Recurrent mode:

To enable the generation of periodic functions, a special mode called "recurrent" is available. If this operating mode is selected, the function will start normally with a start pulse but, when the last value is reached, the generation does not stop and the same function is restarted automatically. Note that in this case, the only way to stop the generation is to apply the stop pulse.

Low jitter mode:

To minimize the jitter between the start pulse and the start of the function generation, the internal CPU can be put into the stop state ("waiting for an interrupt") when it expects the start pulse. The advantage is that the start jitter is 250 ns instead of 1.5 μ s in the other modes. The inconvenience in this mode is that any access to the internal CPU activities is disabled until the end of the function generation.

Fig 2. Different function generation modes

- 1) normal generation. 2) aborted generation 3) event stop/ event start
- 4) new function loaded in the event stop state 5) recurrent mode

DAC mode:

The function generator acts like a normal DAC, i. e. the origin value V0 of the previously selected function is sent to the DAC.

Test mode:

Some test facilities are available in this mode but the access is reserved for the hardware specialists. For further details refer to the GFAS Technical Manual.

5. VME interface

The GFAS module is seen on the VME bus level as an A24:D16:D08(E0) memory interface [6], and responds to address modifiers 3E, 3D, 3A, 39 (hexadecimal). The total memory size is 256 kbytes which can be placed anywhere in the 24 bit address space with jumpers. Since the module contains two function generators, the memory is divided into two blocks of 128 kbytes each. The lower block is dedicated to the channel 0 and the higher to the channel 1. The memory is shared between the VME bus and the two CPUs inside the module. Thus an arbiter (round Robin type [6]) has been implemented to arbitrate the access. The worst case access time for the VME bus is less than 1.5 μ s (the two internal CPUs work at 16MHz).

6. Output format

The GFAS produces at every step a 16 bit word corresponding to the instantaneous value of the function. This value is made available to the user in four different ways:

1) 16 bit parallel output + strobe, 2) analog output with an optional 16bit DAC on a piggy back card, 3) serial opto-coupled link on a screened twisted pair with a maximum transmission length of 70 m, 4) serial optical fiber link with a maximum transmission length of 30 m for plastic fibers.

1) and 2) are not insulated from the VME ground, 3) has an insulation of 2.5kV, and 4) has an insulation in the order of 100kV depending on the length of the optical fiber.

7. DAC (Digital to Analog Converter) card

There are 4 types of DAC cards available now, in different standards. The least expensive one is the single Europe type which can be equipped with an AD669 in the low cost version, or with an AD1139 in its high

precision version (both are 16 bit DACs from Analog Devices). Both versions can be configured for an output of 0 to +10 V unipolar or ± 10 V bipolar. The version equipped with the AD1139 is also able to produce an output of 0 to +5 V unipolar and ± 5 V bipolar. The values sent from the GFAS are also available in digital parallel format on the DAC card. This can be useful if the function is needed in digital form remotely or separated from the VME ground.

8. Software access

Loading of a function is done over the VME bus. The user loads a function through a piece of software called Equipment Module which checks the format and number of vectors and chooses the correct addresses. In the CERN environment, different functions are generated in the different accelerator cycles. Function loading is asynchronous to the function generation. The selection of the function which is going to be generated in the next machine cycle is decided by a real-time program and depends on a telegram received before the start of every cycle. This real-time program is roughly synchronized to the cycle. Precise synchronization with the accelerator cycle is obtained using the start pulse, which immediately starts function generation (with a jitter of 1.5 or 0.25 μ s depending on the mode).

9. Conclusions

The very low price of the GFAS is mainly due to the use of the same CPU for function generation and communication with the VME bus. In addition, because of the serial transmission of the momentary digital function value, the GFAS is easily adapted to a wide range of applications.

Acknowledgment

We would like to thank G. Coudert, P. Lienhard, G. Molinari, J. Philippe, U. Raich and E. Tanke (all CERN), for their support in sharing the knowledge about function generation.

References

- [1] R. Debordes, GFASTAR, CERN/MPS/CO/Note73-18
- [2] J. Philippe, GFAD, CERN/PS/Note 92-12.
- [3] P. Lienhard, LEAR GFD, priv. comm..
- [4] G. Mugnai, MUGEF, CERN/SPS/Note 85-07.
- [5] B. Frammery, CERN/PS/OP/Note 93-16.
- [6] W. D. Peterson, the VMEbus Handbook 2nd edition, p. 23, 27, 89.