IEEE Transactions on Nuclear Science, Vol. NS-28, No. 3, June 1981

A MICROPROCESSOR BASED FUNCTION GENERATOR FOR SUPERCONDUCTING DIPOLE CORRECTION COILS

D. Beechy, D. Bogert, S. Segler, and T. Watts
Fermi National Accelerator Laboratory*
Batavia, Illinois 60510 U.S.A.

Introduction

The design for the Fermilab Tevatron includes approximately 200 dipole correction coils which serve to counteract the dipole field errors of the main superconducting dipole magnets. The current in these correction coils is to be individually controlled by means of 200 separate microprocessor-based function generators which drive separate bipolar 50 amp power supplies. The output of each of these function generators is a correction voltage of 12-bit resolution, calculated by the microprocessor, of the form $v_c(i,t) = i[g(i) + f(t)]$, where i is the instantaneous current in the main superconducting bending dipoles and t is real time. The function generators also have provision for controlling the "on/off" and "reset" functions of the power supplies as well as monitoring 10 status lines from the power supplies. These provisions allow the power supplies to be controlled remotely by the Tevatron host computer via the function generators.

Correction Function Generation

The function generator requires that both the functions g(i) and f(t) be parameterized by the host computer in the form of piece-wise linear segments. Specifically, both g(i) and f(t) may each contain up to 31 linear segments which define the functions. These linear segments are transmitted to the function generator as pairs of numbers, each pair representing the end point of a particular linear segment. For the g(i) curve it is necessary to transmit to the function generator 32 values of i and 32 corresponding values of g(i) if all 31 of the allotted segments are used. The f(t) curve is slightly different in that the function is assumed to begin at t=0 and, therefore, while 32 values of f(t) are sent to the function generator, only 31 values of t need to be sent.

In actuality, values of Δt are sent rather than values of t in order to provide for longer cycle times. The Δt values are up to 16 bits in length with the LSB representing 1 ms. This provides a maximum Δt for each of the 31 possible segments of approximately 65 seconds or a total possible cycle time of something over one half hour.

The microprocessor generates the correction function by sampling both t and i at a 1 kHz rate and by linearly interpolating between segment end points for both the g(i) function and the f(t) function in order to find the corresponding values for g(i) and f(t). These values of g(i) and f(t) are then summed together and multiplied by i to produce the correction function, v_c. The final output stage is a 12-bit DAC which converts the correction function to a corresonding analog signal.

The particular form of the correction function, $v_c = i[g(i) + f(t)]$, in which the output of the function generator depends on the real time main dipole current was chosen for several reasons. First, the i dependence means that the DAC output will automatically scale with the current if it changes for any reason. In addition, explicitly multiplying by i allows the functions, f(t) and g(i), to be fairly simple. The f(t) function is mostly used for operations which are time dependent during the Tevatron cycle such as injection or extraction. For these cases, multiplying by i ensures that the time dependent part of v_c scales to first order with the proton energy.

The segment end point values for g(i) and f(t) are transmitted to the function generator as binary integers of up to 15 bits, which may at first seem unnecessary since the final output is to only 12-bit resolution from the DAC. This apparent disparity in the required resolution for g(i) and f(t) is a result of the necessity of making the output fully variable over the full scale range, both at injection ($i \approx 1/8$ of full field) and at full field. The values of i are binary integers of up to 12 bits. Normally the result of multiplying a 12-bit integer by the 15-bit sum of g(i) and f(t) is a 27-bit result, the top 12 bits of which are loaded into the DAC. However, at injection the value of i drops to only 9 bits and the above multiplication now produces 3 leading zeros which are loaded into the 3 most significant bits of the DAC. In other words, the output range of the DAC is restricted to 1/8 of its full-scale output at injection. If, however, the top 3 bits of the result of the $i[g(i) + f(t)]$ multiplication are ignored, and the next 12 bits sent to the DAC, full output from the DAC is again possible even at injection. However, this maneuver requires caution since it is now possible to "overflow" the DAC at i values above injection. The functions g(i) and f(t) must be judiciously chosen so as to ensure that for currents above injection, the summation of g(i) and f(t) is not so large as to produce a product in which the 3 most significant bits are anything other than zeros. However, within this restriction, it is then possible to have the DAC output any value within its full-scale range both at injection and at full field.

Hardware

The function generator is contained in a single width Camac module and communicates with the host computer via the Camac system of the Tevatron. Additional one-way commands from the host to the function generator arrive via the Tevatron clock which is a single serial signal that is encoded to carry commands and timing information to the various Tevatron systems. This clock signal enters the function generator via the rear I/O connector, and is decoded on the function generator board by a custom hybrid circuit board which was designed by Fermilab. Interconnections between the function generator and the power supply that powers the correction coil are also made through the rear I/O connector. In addition, the 12-bit digital value of "i," is presented to the function generator via the rear connector. The front panel of the generator contains three LEDs which give a limited visual status check of the card. These three LEDs are the customary "N" light to indicate a Camac dataway cycle to the function generator, a "Ramp Enable" LED which indicates that the host computer has authorized the function generator to output an analog

*Operated by Universities Research Association, Inc. under contract with the U. S. Department of Energy.

0018-9499/81/0600-2311$00.75©1981 IEEE

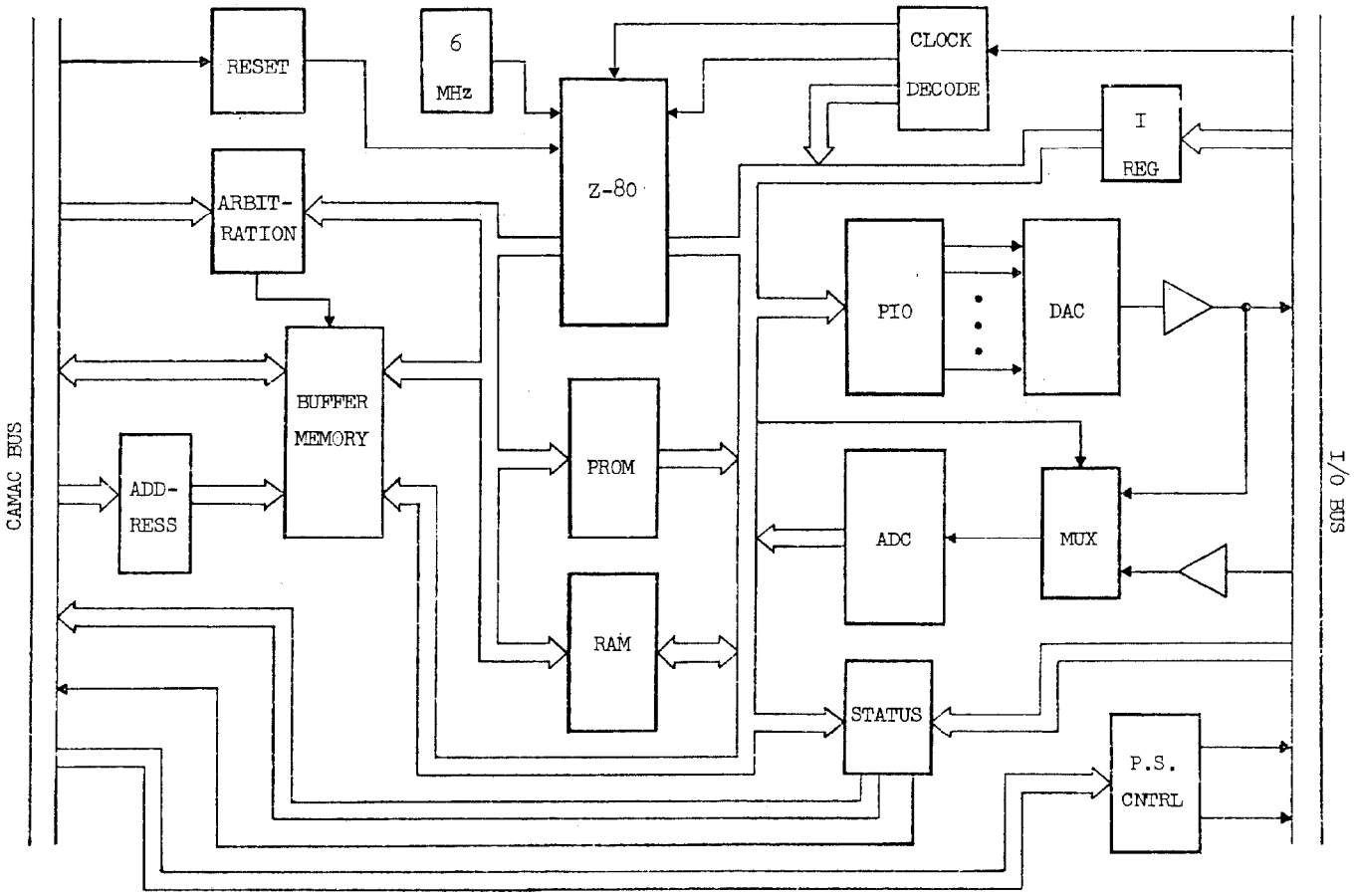

Fig. 1. Block diagram of the function generator.

signal, and a "heartbeat" LED which is functional only if the microprocessor is executing the program stored in the on-board PROM.

The basic components of the function generator are shown in the block diagram of Fig. 1 and are discussed below.

Microprocessor, PIO, Memory

The microprocessor is the Zilog Z-80B which is an 8-bit machine and can use an input clock frequency of up to 6 MHz. This high clock frequency proved necessary in order to provide for the relatively large amount of arithmetic which must be accomplished at a 1 kHz rate. This processor also provides a convenient vectored interrupt scheme which allows individual interrupts to direct the processor to the corresponding service routine in the program. Interrupts to the processor are described later in this paper.

The program for the processor resides in a single 4-kbyte, 2732-type EPROM which provides for relatively easy changes to the program as may be necessary, or desirable in certain specialized applications. 1 kbytes of read/write memory are also provided for scratchpad calculations in generating the correction function and for storing the parameters of the g(i) and f(t) functions. The Zilog PIO is used as an interface between the microprocessor and the 12-bit DAC.

Buffer Memory

The buffer memory is used by both the host computer, via the Camac system, and the microprocessor for communications between the two. Specifically, it is to the buffer memory that the host computer loads the values of the parameters which describe both the g(i) and the f(t) functions. The buffer is configured to appear to the Camac system as 128 16-bit words and as 256 8-bit words to the microprocessor. In writing to the buffer memory, the Camac system has only to specify a starting address in the buffer memory by the use of an F(20) command and then sequential 16-bit words may be written into the buffer with the F(16) command. An automatically incremented address pointer on the function generator places these 16-bit words in sequential memory positions. The same is true if the Camac system reads the buffer memory--again, only a starting address is required followed by sequential "reads" from the card with the Camac F(0) command. To the microprocessor, the buffer memory appears as an ordinary block of read/write memory with the exception that an arbitration flip flop must be set before accesses can be made to the buffer. This communications scheme allows for rapid down-loading of new parameters from the host computer to the function generator since the microprocessor is not involved with the transfer and thus no "handshaking" between the host and the microprocessor is required.

The arbitration logic for the buffer memory is responsible for preventing errors in the buffer in those situations when one system has gained access to the buffer and the other system attempts an access before control of the buffer has been relinquished. Essentially, control of the buffer memory is granted by the arbitration logic on a first come, first-served basis. For the Camac system, control of the buffer is granted on a word-by-word basis and is automatically relinquished after each read or

write to the buffer. The microprocessor, however, is able to set a control flip flop which guarantees access to the buffer for as long as this flip flop is set. Because the buffer is small and accesses to it are at high speed, conflicts between the Camac system and the microprocessor for buffer control are rare. For the case when the Camac system attempts to access when the microprocessor has control of the buffer, the function generator will return a "no-Q" to the crate controller and ignore the Camac command. The microprocessor is informed that the buffer memory is unavailable when it attempts to set the above-mentioned control flip flop. If the buffer is unavailable, the processor will be unsuccessful in setting this flip flop and a subsequent "read" of the flip flop will inform the processor that it is still reset and that control of the buffer has not been granted.

ADC, Multiplexor

A 12-bit A/D converter is used on conjunction with a two-input analog multiplexor to monitor the analog output of the function generator and an analog signal from the correction coil power supply which is proportional to the current in the coil. The multiplexor is controlled by the microprocessor which, on command from the host computer, reads the ADC and writes the converted digital values for both the above signals into the buffer memory where they are available to the host.

Interrupts

The microprocessor keeps track of real time by means of interrupts at the Non-Maskable Interrupt input of the processor which occur at a 1 kHz rate and are derived from the Tevatron clock. Additionally, a vectored interrupt scheme is used with the Tevatron clock decoder to allow the host computer to control the operation of the function generator. These interrupts are maskable and are time and order dependent. A functional explanation of these interrupts is given below.

t-Stop. This interrupt causes the microprocessor to stop calculating new values of f(t) while continuing the generation of g(i). After this interrupt is received, the microprocessor still continues to calculate the correction function, v_c; however, the f(t) value is constant and equal to the value calculated just prior to the t-STOP interrupt.

t-CONTINUE. This interrupt is required to follow the t-STOP interrupt and commands the processor to again develop values of f(t) where it left off.

t-NEW. This interrupt commands the microprocessor to access the buffer memory and to copy the function parameters in the buffer into the read/write memory of the processor. The processor then begins immediately to generate a new g(i) function based on the new parameters; however, generation of a new f(t) function is not begun at this time, but rather f(t) is given an initial value as specified by the new parameters.

t-START. This interrupt causes the microprocessor to begin the development of the f(t) function based on the parameters copied into memory under the t-NEW interrupt.

t-READ. This interrupt is a command to the processor to write into the buffer the most recently calculated values of g(i), f(t), and v_c as well as the most recently sampled value of i.

t-Double Value. The g(i) function is actually constructed of two sets of parameters, the first of which is associated with the injection, acceleration, and flattop portions of the Tevatron cycle while the other set of parameters is used for the extraction portion of the cycle. This interrupt signals the microprocessor to switch from one set of parameters to the other.

Modes of Operation

There will be two modes of operation for the function generator when used with the Tevatron, Pulsed and DC. These modes depend entirely on the sequence of the above timing interrupts.

The **Pulsed** mode is the "normal" mode for the Tevatron and will be used whenever the machine is executing ramped acceleration cycles. In this mode, changes to the g(i) and f(t) functions are made only on a cycle to cycle basis, although parameters for a new function may be down-loaded to the buffer memory at any time during a machine cycle. Not until the end of extraction, when no beam is in the machine, is the t-NEW interrupt issued to the processor. Sometime later, before injection of the next cycle, the t-START command is issued causing the new correction function, v_c, to be started. In addition, the function generator can be made to stop development of the f(t) function at any value of f(t), by other than a t-STOP interrupt, by setting a stop bit in the associated f(t) parameter when it is down-loaded from the host to the function generator. The processor, upon encountering the stop bit, will stop the development of f(t) when the selected value of f(t) has been reached.

For the case when the Tevatron contains stored beams, the **DC** or **Storage** mode of the function generator is used. In this mode, great care must be taken in changing function parameters to insure that no discontinuities are introduced into the correction function with attendant loss of beam. To this end the following sequence of commands will be issued to the function generators. A STOP command is given causing all generators to hold to their present values of f(t) and g(i), presuming that i is unchanging in storage mode. The host then issues a READ command and subsequently accesses the buffer memories of those generators where new parameters are to be written in order to ascertain the most recent values of g(i) and f(t). The host then calculates new parameters making sure that these parameters introduce no discontinuities into v_c when the subsequent NEW command is issued. The final command, START, causes execution to begin the new f(t) function. This development continues until another STOP command is received or a stop bit is encountered in the f(t) parameters.

Acknowledgments

The authors wish to acknowledge the technical assistance of R. Williams and C. McClure for their efforts in the construction and debugging of the prototype units of the function generator.