Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

Design and Implementation of Digital Missile Trajectory Controller

By

A.F.Sobh*[1], A.M.Eldamarawy**[2], Y.Z.Elhalwagy***[3], R.A.Elbardeny****[4]

Abstract

The missile mission success depends on robust guidance techniques in which the missile is to be guided to fulfill a correct trajectory up to hit point. This paper is devoted to discuss the design, implementation and verification of a digital trajectory generator (DTG) and its application for guided vehicles. The DTG is fed with target position and environmental parameters. The suitable trajectory missile attitude angles are generated as a function of time and are transmitted from the launcher to the missile via the four standard control wire dispended from the missile during flight. For the simulation purposes, an adaptation is developed to interface the DTG with the software based missile simulator through the standard USB port. A synchronizer is embedded in the DTG hardware design to synchronize and simulate the missile leave launch event. The design is verified by comparing the results of the basic stand alone flight simulator and those with the DTG hardware in the loop simulations in various operating points. Promising results showed potential for an on-going research and development for the system of concern. Developed DTG is a must step on upgrading the guided vehicles via using a goniometer in the launcher or via integrated onboard with homing head.

Keywords

Robust guidance, Digital trajectory generator (DTG), Embedded Systems, Programmer electronic controller (PEC), semi-automatic command to line of sight (SACLOS).

[1] * M.Sc. student, Dpt. Of Guidance, Military Technical College, Cairo, Egypt.
[2] ** Ph.D. Defense Systems Studies Center, Cairo, Egypt.
[3] *** Associated Prof., Guidance Dep., Military Technical College, Cairo, Egypt.
[4] **** Ph.D., Lecturer, Guidance Dep., Military Technical College, Cairo, Egypt.

I. Introduction

The frequent minor wars have brought to the fore guided missiles as the main weapon against all types of military targets. Any weapon should have as high a single shot kill probability as possible [1]. In semi-automatic command guidance system an operator or a computer at the control point solves the mission of interception on the basis of obtained coordinates for both target and missile and forms the command.

There are some instrumentations to measure the missile behavior which might be contained in the missile itself or suited at a ground or a mobile platform control center. The coordinates of both missile and target are continuously measured and sent to the guidance computer that evaluates the necessary corrections for the missile flight. Based on the appropriate guidance method, the computer is able to determine what maneuvers the missile should execute in order to improve its chances of hitting the target. The computer passes steering instructions, such as desired lateral accelerations in the pitch and yaw planes, to the control system (autopilot), which moves the missile control surfaces [2].

One way of restoring the single shot kill probability is to use large warhead with large lethal area, but this will usually mean a larger missile. The other method that can be adopted is to design a robust guidance system to reduce the miss-distance with high single shot kill probability. This can be accomplished by monitoring continuously the flight parameters of the missile and target, then employing this information to control the missile in space [3].

An anti-tank guided missile is a weapon system primarily designed to hit and destroy armoured tanks and other armoured vehicles. Developed first-generation command-guided and second-generation semi-automatic command guided missiles had many disadvantages and lower hit rates. For that reason, third generation imaging infrared fire-and-forget missile concept is very popular [4]. The missile launcher comprises an analogue computer consisting of yaw and pitch channel and subsidiary circuits concerned with timing, shaping and limiting the direct program parameters. These parameters are used to direct the missile towards the controller's sight to target and then of constant demands which hold the flight path parallel to, or directly down the sight line. The timing of the shaped demands is being such that they hold when the flight path and sight line are almost coincident. Manual demands from the gunner are superimposed upon the programmed levels after "take command" enabling the operator to adjust the missile flight path onto target. This kind of missile guidance is shorted as semi-automatic command to line of sight (SACLOS) [5], [6], [7].

A ground target tracker used to implement the semi-automatic command to line of sight (SACLOS) law in initial and midcourse phases employs a passive sensor such as a T.V.

camera or an IR localizer with wide field-of-view for missile capture after launch [4], and a differential tracking system with narrow field-of-view for target tracking. The SACLOS guidance law is of a three-point guidance type such that the missile stays almost on the line-of-sight (LOS) to the target after missile capture. Therefore, missile-to-target relative position vector information is available from a single ground tracker.
The programmer electronic controller (PEC) consists of two separated channels one for yaw and other for pitch. The yaw and pitch demands direct the missile towards the controller's sight to target. When a firing sequence is initiated, the PEC starts to function, but due to the inhibit signal; its outputs are held at zero demand level ensuring that the jetevator thrust remains parallel with the launcher box rails.
Constant growth of computational power demand requires the multiprocessor computing structures to advance their internal mechanisms. Such architectures include many, up to millions, processors able to perform processing tasks. Each technique increasing the efficiency results in the significant benefit regarding operational energy and time of task execution. Due to large scale of multiprocessor computing structures, the importance of achieving faster and efficient systems is invaluable. Such approach is promising for large scale computations for embedded system applications for guided vehicles [8].
In this paper, digital computer hardware is developed for fulfilling the same analogue counterpart. The developed DTG is considered as a basic part in an upgrading regime for the SACLOS technique by integrating it onboard the missile. The reminder of this paper is organized as follows: Section II presents modeling and design for the proposed DTG associated with the carried out software. Section III contains a description of simulation and experiments result analysis. Conclusions and final remarks are presented in Section IV.

II. Modeling and design

This section is devoted to present the design of proposed programmer electronic generator associated with the necessary hardware and software developments. For purpose of verifications, a generic six-degrees-of-freedom flight simulation model for flying vehicle has been carried out. The PEC is an analogue computer consisting, basically, of yaw and pitch channels and subsidiary circuits concerned with timing, shaping and limiting the direct and separated program parameters. The yaw and pitch demands are in the form of DC currents, fed to the missile via a 4 wire link in the missile control wire. These demands are programmed so that initially they consist of shaped demands which direct the missile towards the controller's sight -to target and of constant demands which hold the flight path parallel to, or directly down the sigh line, the timing of the shaped demands being such that they terminate when the flight path and sight line are almost coincident. Figure (1) represents closed loop s-Function with conventional programmer electronic generator. The 6DOF simulation model runs on the

host computer. The pitch and yaw programs are functions in many variables (initial missile angles, time, and temperatures, etc) that affects the missile path during flight, and they are implemented in the mathematical equations (1) and (2) for pitch and yaw plan respectively:

$$\theta_p = -\left(\theta_A - \theta_B\, e^{-\frac{t}{\tau\theta}}\right) \quad (1)$$

Where

$$\theta_A = k_1 - (\theta_s - k_2)$$

$$\theta_B = E^0 + k_3\theta_s$$

T_0=2.5 seconds; E^0 is temperature compensation term; K_1, K_2, K_3 are desired values.

$$\Psi_p = (\Psi_A + \Psi_B)\left(1 - e^{-\frac{t}{\tau\Psi}}\right) \quad (2)$$

$$\Psi_A = \Psi_s$$

$$\Psi_B = k_4\Psi_s$$

$$(\Psi_A + \Psi_B) = 1.3\Psi_s \quad \text{For } t > t_0 > t_y$$

K_4 desired value, t_0= box brake wire, t_y= end of 'program to sight-line' phase of engagement.

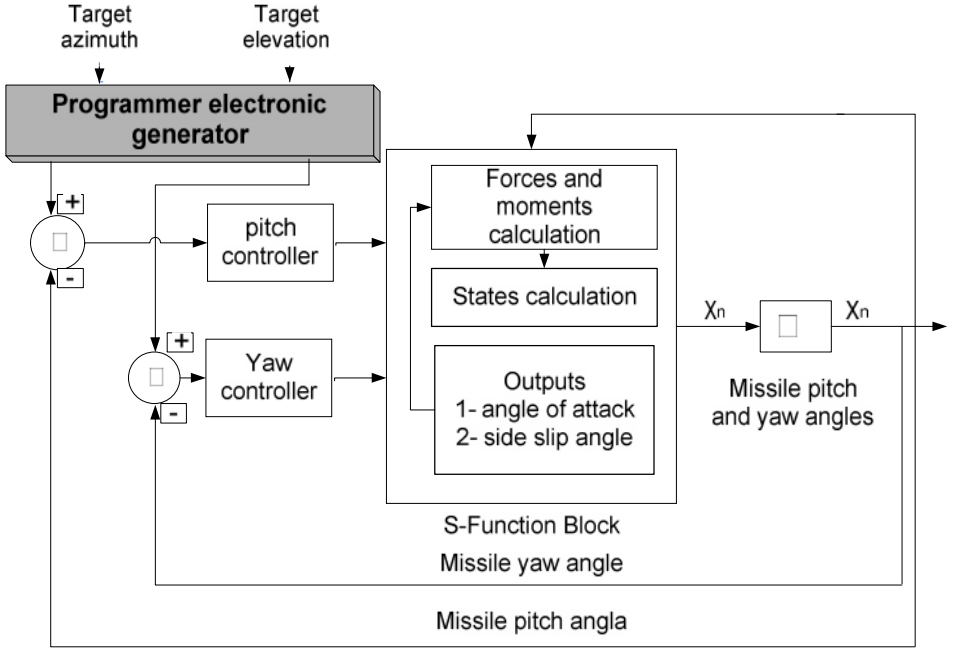

Figure (1): Missile closed loop S-Function with programmer electronic generator hardware

The design entry comprises hardware DTG, a communication channel, and a personal computer running a missile simulator on MATLAB [9] as shown in Figure (2). The code developments are utilized using C-programming language associated with Atmel microcontroller and the missile simulator S-function. The communication channel utilizes a standard USB port on both sides.

The DTG hardware was designed using ATmega16 with a Harvard AVR core which provides multiple features such as a 16K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain [10].
The first operational stage utilizes five analog-to-digital conversion (ADC) inputs mapped on the microcontroller's PORTA. These inputs are, Ψs represent target position in azimuth, θs represent target position in elevation plane, £ crest angle, Ψd joy-stick guidance demand in azimuth, and θd joy-stick guidance demand in elevation plane.
The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents the power supply return voltage and the maximum value represents the voltage on the AREF pin. Optionally, AVCC or an internal 2.56V reference voltage may be connected to the AREF pin by writing to the REFSn bits in the ADMUX Register. The internal voltage reference may thus be decoupled by an external capacitor at the AREF pin to improve noise immunity.
PORTB pins in the microcontroller are used as discrete inputs. A push button simulates the leave launch trigger for starting the guidance demands, and a toggle switch represents the take command. Those inputs are used in programming as shown in the software to affect the discrete values in the communication frame transmitted to the flight simulator. In the MATLAB flight simulator Simulink block called serial receive is added, which receives the frame transmitted by DTG to carry out the required communication tasks in receiving serial data over the USB port.

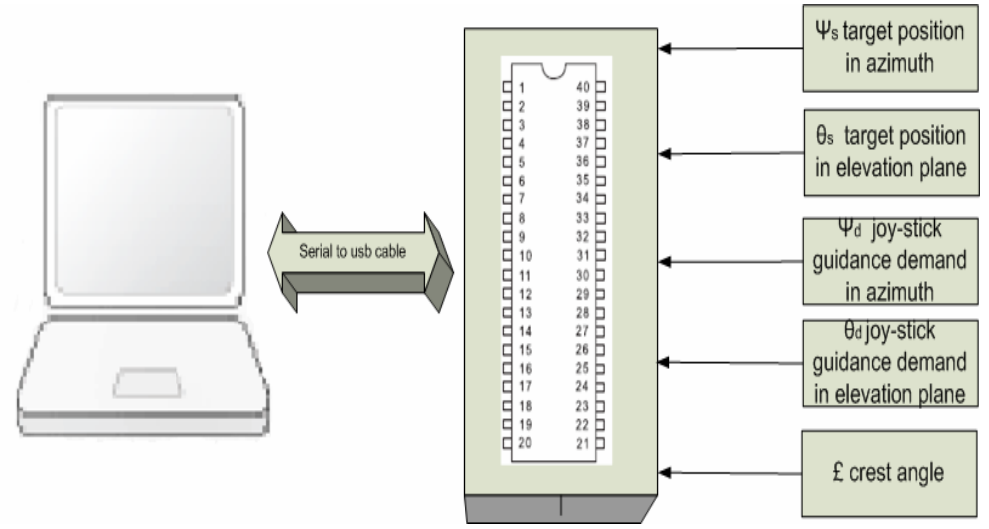

Figure (2): Hardware in the loop facility

The utilized C language appendix (I), software is divided into a set of functions for initialization and performing calculations and timers by which the microcontroller produces a new data set every 10msec. This new updated demand values are packed in a communication frame and made ready for transmission using TXFRAME. The correct values and attitude angles calculations are carried out in the PEC function. The ADC registers are programmed to convert a channel every 104usec, thus MATLAB flight simulator can obtain a complete set of new data every 520usec. The code script is found in appendix (I).

III. Simulation results and verification

The DTG design was verified by comparing the results of the software stand alone flight simulator and those with the DTG hardware in the loop simulations in various operating points. Simulation outputs for the two cases are saved from Simulink's work space. The Outputs are compared with each other to give contrast about the degree of matching between the two runs for pitch demand and missile trajectory.

As shown in Figure (2) and Figure (3), the pitch demand generated from the software block in Matlab (simulink) adheres to that generated by hardware (DTG), and that the trajectory output of the two demand sources are almost similar, which means that the developed DTG satisfies the same operating parameters needed in the real time simulation regarding that the DTG is using the onboard synchronizer that satisfy the real time operation.

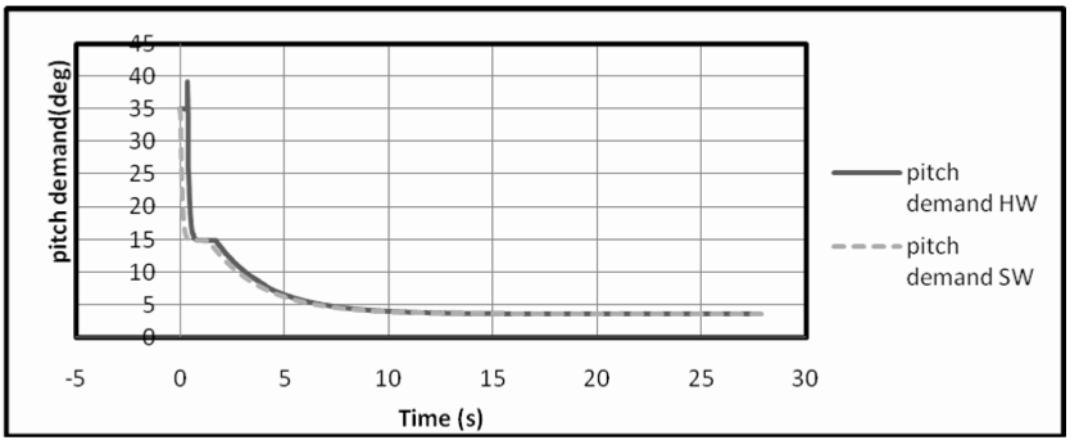

Figure (3): Software and hardware pitch demands

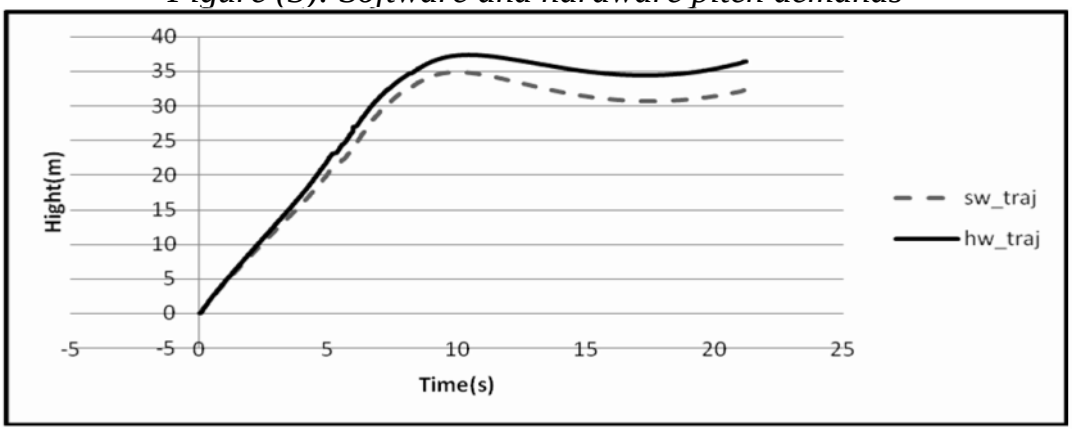

Figure (4): Software and hardware trajectories

Promising results showed potential for an on-going research and development for the system of concern. Developed DTG is a must step on upgrading the guided vehicles. Comparison study showed satisfactory results. The developed digital computer fulfilling the analogue counterpart parameters can be used in the upgrading regime for the SACLOS technique by integrating it onboard the missile.

IV. Conclusion

This paper addressed an important issue of command guidance missile trajectory generator. Investigation of embedded system implementation for DTG is provided. In particular, developed digital computer fulfilling the same analogue counterpart parameters is important step on upgrading the first generation antitank missile to a second or third generation and can be used in the upgrading regime for the SACLOS technique by integrating it onboard the missile.

References

[1] Eaves J.L.; "Principles of modern radar", Van Nostrand Reinhold company Inc., (1987).
[2] Garnell P. and East D. J., "Guided Weapon Control Systems", 2nd edition, Pergamon press, New York, 1980.
[3] Locke S. and East D.J., "Principles of Guided Missile Design", Van Nostrand Reinhold company Inc., 1955.
[4] Song T. L. and Um T. Y., "CLOS+IRTH Composite Guidance", IEEE Transactions on Aerospace and Electronic Systems, vol. 33, N0. 4, 1997.
[5] Mohsen S. Aly; "Liner model evaluation of command guidance systems" proceedings of the 8th Int. conference on aerospace science and aviation technology, pp1001-1003, 4-6 May, (1999).
[6] Y. Z. Elhalwagy and M. Tarbouchi, "Fuzzy logic sliding mode control for command guidance law design," ISA Transactions, vol. 43, no. 2, 2004.
[7] Oda, A.N., G.A. El-Sheikh, Y.Z. El-Halwagy and M. Al-Ashry, "Robust CLOS Guidance and Control Part-1": System Modelling and Uncertainty Evaluation, 14th International Conference on Aerospace Sciences & Aviation Technology, (2010).
[8] C. Grzegorz, D. Zydek, Y. Z. Elhalwagy, and H. Selvaraj, "Overlay-NoC and H-Phy approaches to computing based on Modern Chip MultiProcessors", IEEE International Conference on Electro/Information Technology IUPUI, Indianapolis, Indiana, USA, May 6-8 2012.
[9] The math works Inc. SIMULINK Toolbox User Guide, Second Edition, USA, 1999.
[10] Data sheet for ATMEGA16 microcontroller.

Appendix(I)

Sample initialization codes:

```
void port_init(void)
{ PORTA = 0x00;  DDRA    = 0x00;
 PORTB  = 0x00;   DDRB    = 0x00;
 PORTC  = 0xFF;   DDRC   = 0x00;}
{ ADCSR = 0x00; //disable adc
 ADMUX = 0x00; //select adc input 0
  ADCSR = 0xCF;}
```

pragma interrupt_handler adc_isr:15

```
void adc_isr(void)
{float AEC=0;
ADCBUFF[ADMUX]= (signed int)ADCL;
ADCBUFF[ADMUX] |= (signed int)ADCH<<8;
if (ADMUX ==0x00)
{AEC = (float)Azimuth;  AEC -= 511;  AEC *= 1.5655577; PsiS=AEC;}
if (ADMUX ==0x01)
{AEC = (float)Elevation;  AEC -= 511;  AEC *= 0.69667318;  ThetaS=AEC;}
if (ADMUX ==0x02)
{AEC = (float)CrestAngle;  AEC -= 511;  AEC *= 0.391389;  crest=AEC;}
if (ADMUX ==0x03)
{AEC = (float)LR;  AEC-=JoystickLROFF;  AEC *= -0.69471;  JoystickLR=AEC;}
if (ADMUX ==0x04)
{AEC = (float)UD;  AEC -= JoystickUDOFF;;  AEC *= -0.69471;  JoystickUD=AEC;}
ADMUX  ++;
if (ADMUX == 0x05)   {ADMUX=0x00;}
ADCSRA |= 0x40;}
void timer1_init(void)
{ TCCR1B = 0x00; //stop
 TCNT1H  = 0x63;   TCNT1L  = 0xC1;
 OCR1AH = 0x9C;  OCR1AL  = 0x3F;
 OCR1BH = 0x9C;  OCR1BL  = 0x3F;
 ICR1H     = 0x9C;  ICR1L       = 0x3F;
 TCCR1A  = 0x00;   TCCR1B  = 0x0A; //start Timer}
```

pragma interrupt_handler timer1_compa_isr:iv_TIM1_COMPA

```
void timer1_compa_isr(void)
{if (((CTRL & LEAVE)==0x00)&(tprog<30))
{tprog +=0.02;
if ((PINC & TakeComm)==TakeComm)
    {JoystickLRINT+=JoystickLR*0.02;JoystickUDINT+=JoystickUD*0.02;}
```

```
PEC();TXFRAME();}
else;{TXFRAME();}
if ((CTRL & LEAVE)==LEAVE)
{tprog=0;JoystickLRINT=0;JoystickUDINT=0;ThetaD=0;PsiD=0;JoystickLROFF=(flo
at)LR;
  JoystickUDOFF=(float)UD;}}
```

Sample computation functions

```
void PEC(void)
{tp=t0 + 1.37;
Tinc = t0 + 0.95;  // T_incidence must be less than ty
PsiD1 = 1.3 * PsiS;
if (PsiD1 >= 1067)
 {ty = 1.5 + (PsiD1 - 1067) * 0.0045; PsiD1 = 1067;}
else  ty = 1.5;
```

Pitch program

```
 if (tprog < t0 )  {ThetaD= (E+0.58*Y0)*0;}
 if ((tprog >= t0) & (tprog<tp))
 {ThetaD= (1-exp((t0-tprog)/taoBoost));
ThetaD *= (-557 + ThetaS);
ThetaD+= (E + 0.58*Y0);
ThetaD+=JoystickUDINT;       }
if (tprog >= tp)
{ThetaD= (-557 + ThetaS) + (E + 0.58*Y0)*exp(-1*(tprog-tp)/taoComp);
ThetaD+=JoystickUDINT;       }
if ((ThetaD+ 622) >= 444)  {InRange=1;}
if ((ThetaD+ 622) < 444)    {InRange=0;}
```

yaw progran

```
if(tprog<t0)PsiD=;
 if(tprog>=t0&tprog<Tinc)
{if(InRange==1)PsiD=(PsiD1*(1-exp(-1*(tprog-
t0)/taoPsi)))*(cos((ThetaS+36)*MIL2RAD)*PsiFac);
if(InRange==0)PsiD=(PsiD1*(1-exp(-1*(tprog-t0)/taoPsi)))*(cos((ThetaS+
36)*MIL2RAD));}
if(tprog>=Tinc&tprog<ty)
{if(InRange==1)PsiD=(PsiD1*(1-exp(-1*(tprog-t0)/taoPsi)))*(cos((ThetaS+
36)*MIL2RAD))*PsiFac;
if(InRange==0)PsiD=(PsiD1*(1-exp(-1*(tprog-t0)/taoPsi)))*(cos((ThetaS+
36)*MIL2RAD));}
if(tprog>=ty)PsiD=(PsiD1*(1-exp(-1*(tprog-t0)/taoPsi)))*cos((ThetaS+
36)*MIL2RAD)-(0.3*PsiS*(1-exp(-1*(tprog-ty)/taoinc))); PsiD+=JoystickLRINT;  }
```