INTEGRATION OF AN ELECTRICAL DISCHARGE MACHINING MODULE ONTO A RECONFIGURABLE MACHINE TOOL

B.H. ROBERTS

2014

INTEGRATION OF AN ELECTRICAL DISCHARGE MACHINING MODULE ONTO A RECONFIGURABLE MACHINE TOOL

By

Bryndan Roberts

Submitted in fulfilment of the requirements for the degree of Master of Engineering in Mechatronics, in the Faculty of Engineering, the Built Environment and Information Technology, to be awarded at the Nelson Mandela Metropolitan University

2014

Supervisor: Prof IA Gorlach

Co-Supervisor: Prof TI van Niekerk

Nelson Mandela Metropolitan University
for tomorrow

DECLARATION

I, Bryndan Roberts (s208092273), hereby declare that the dissertation for Master of Engineering in Mechatronics is my own work and that it has not previously been submitted for assessment or completion of any postgraduate qualification to another University or for another qualification.

Bryndan Roberts

Table of Contents

List of Figures

Nomenclature

CNC	Computer Numeric Control
EDM	Electrical Discharge Machining
HMI	Human-Machine-Interface
HSM	High-Speed Machining
RMS	Reconfigurable Manufacturing System
RMT	Reconfigurable Machine Tool
TDM	Tool, Die and Mould

Abstract

Electrical Discharge Machining (EDM) is a non-contact manufacturing process in which material is removed from a metal workpiece by high frequency electrical pulses produced between an electrode and the workpiece. EDM machines are usually stand-alone devices, and are quite expensive. The objective of this research was to integrate an EDM machine and an existing reconfigurable CNC machine tool, using a modular approach, to enable conventional milling and EDM to be conducted in a co-ordinated fashion on the same machine tool.

In an EDM process, feedback control is crucial. There is continual measurement of conditions at the tiny gap between the electrode and the workpiece; this measurement determines the movement of the electrode (by means of some actuator), in order to maintain the correct sparking conditions. For ideal integration, the path that the electrode must follow should be determined by the CNC machine tool controller, but the advancement along this path should be dictated by the EDM machine controller. Thus some form of communication between the two machines is required.

A variety of integration strategies were implemented, and their performance evaluated. These included using the EDM machine's controller to drive the CNC machine's servo motors directly in DC mode, and using the control signal from the EDM machine to communicate with the PC based CNC control program, to achieve advancement and retraction of the electrode using the CNC machine's servo motors. The main focus of the investigation was however the use of an external controller, which would effectively replace the CNC machine's controller, and allow a small linear actuator mounted on one of the CNC machine's axes to produce fine control of the electrode movement. The external controller was implemented in LabVIEW, using a data acquisition device. Also, PC simulation models of the actuator, controller, and EDM process were produced.

Keywords: *Electrical Discharge Machining, EDM, Control Systems, CNC, Reconfigurable Machine Tool*

1. Introduction

In past years, there has been a steady decline in the tool, die and mould (TDM) manufacturing industry in South Africa. This has fortunately been recognized by the South African Department of Trade and Industry, who formed the SA TDM Incentive, which aims to reduce the import of tools, dies and moulds into South Africa, in favour of local manufacture, and to improve SA's TDM competiveness in the global market [1].

This research forms part of a collaborative project effort between the Nelson Mandela Metropolitan University (NMMU), the University of the North West, and the University of Pretoria. Funded by the Department of Trade and Industry, the aim of the project was to develop a low cost manufacturing system intended for SA SMME's in the TDM industry [1,2]. This project falls within the ambit of Reconfigurable Manufacturing Systems. RMS is a (mainly) system level approach which aims, as opposed to dedicated production systems on the one hand, and Flexible Manufacturing Systems (FMS) on the other, to allow manufacturing systems to be modified by the inclusion of additional modules, as need arises due to changes in product design, for example. At the same time it seeks to be less costly than flexible systems, as functionality is only incorporated into the system when and as necessary, whereas in FM systems, often functionality is provided (and paid for), which is never fully utilized [3]. RMS is based on a modular approach, where functional modules can be either added to a system, or reconfigured (i.e. re-arranged physically and/or functionally), to achieve system capabilities as required by the items being manufactured [4-7]. It is particularly useful in the TDM industry, where new moulds are constantly being produced due to design enhancements of products to be manufactured.

Over the past few years the NMMU has developed a CNC Reconfigurable Manufacturing Tool (RMT) as part of the RMS initiative [1,2]. This machine tool is basically a milling machine, and is also intended to have a modular and reconfigurable nature. The RMT is shown in Figures 1.1 and 1.2 overleaf.

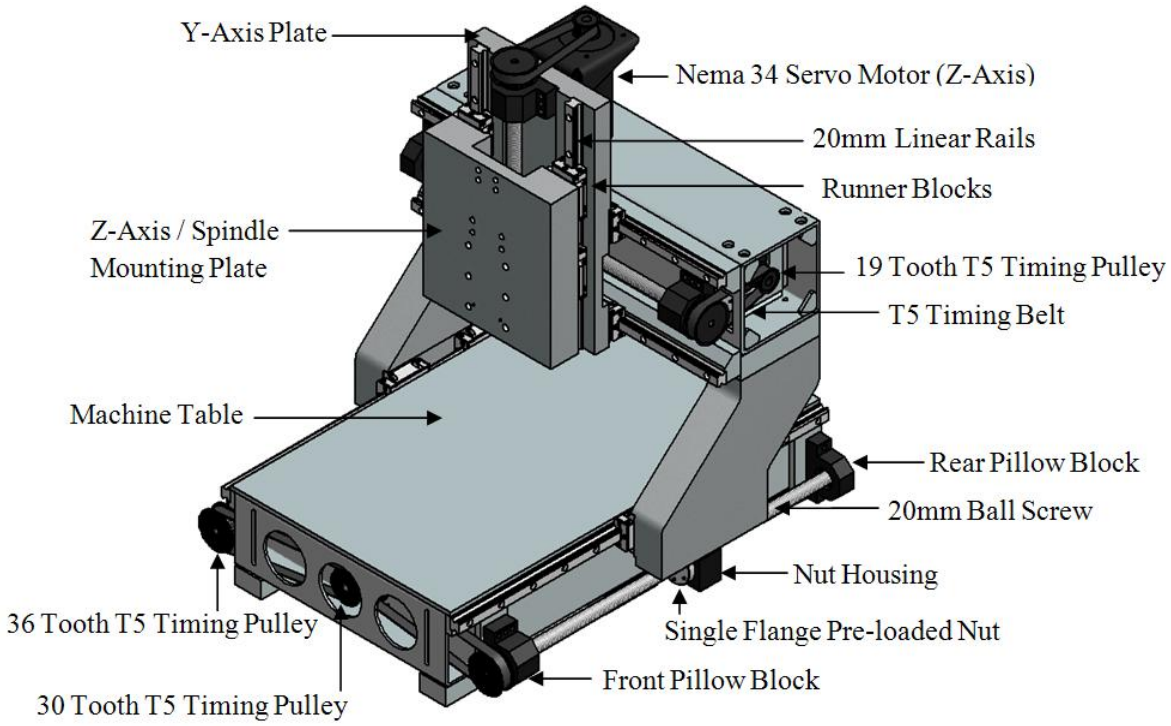


Figure 1.1 Reconfigurable Machine Tool developed by NMMU: Machine Base Structure [1]

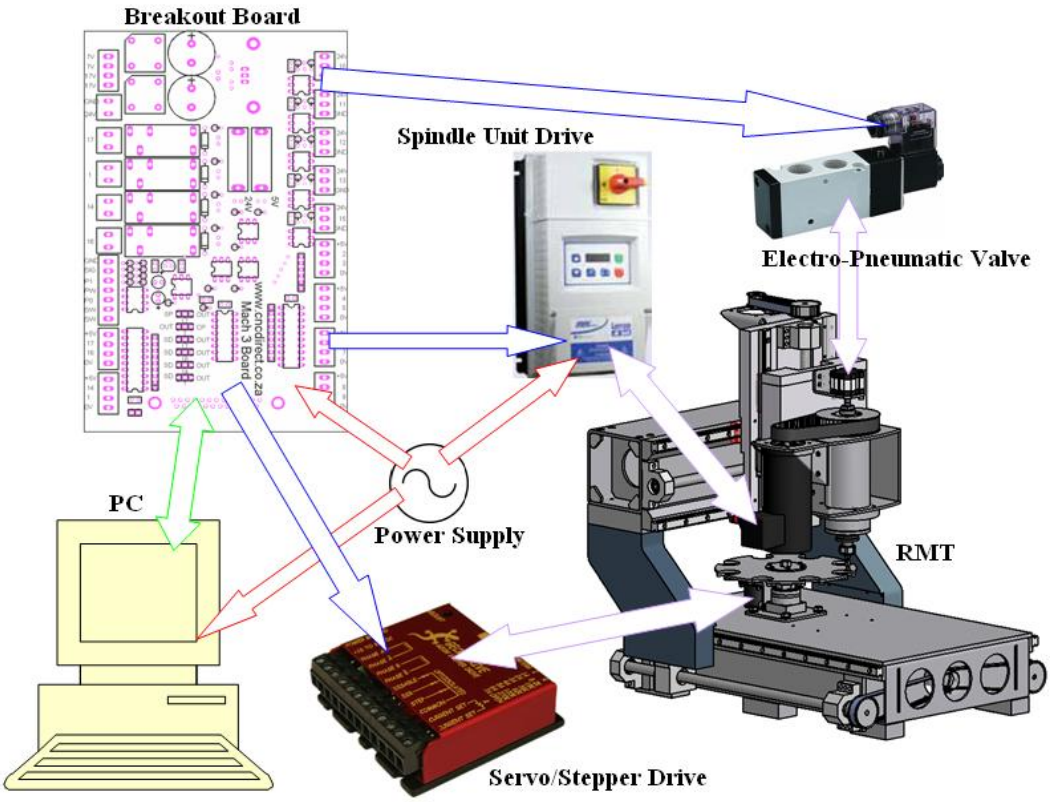


Figure 1.2 Reconfigurable Machine Tool, showing Tool and Control System Overview [1]

The particular project of the present dissertation involves the investigation of the integration of an Electrical Discharge Machining (EDM) module onto the CNC machine tool. As such, it aims to include either an off-the-shelf EDM unit onto the machine, or a new EDM unit could be developed and integrated onto the machine, such that the versatility and capability of the RMT is improved. The EDM process, described below in more detail, is a particularly useful manufacturing technique in the TDM environment, as very hard metals, e.g. hardened tool steels, can be easily machined, whereas such materials are particularly difficult to machine using conventional cutting (milling, turning) techniques. Thus EDM is often used for finish machining after the hardening process in tool and die manufacturing, since the tool or die typically loses dimensional accuracy during heat treatment in the hardening process.

EDM machines are typically stand-alone machines dedicated to the EDM process, and are expensive. Generally they are CNC machines, and are used either for 'sinking' type EDM, or for 'wire-cut' EDM. Parts need to be transferred from other machines to the EDM machine and set-up on the EDM machine. Sinking EDM is most often a single axis process, although multi-axis machines, naturally more costly, do exist. This research investigates the inclusion of a simple EDM device onto the RMT, in an integrated but modular manner, i.e. the device is to from part of the machine's capabilities and its control is to form part of the total machine's control system, such that a manufacturing process can include e.g. milling, and then finish EDM, in a single set-up (thereby also improving accuracy and reducing overall set-up time). Ideally, it is also desired that the EDM process can be expanded to multi-axis capability, by using the CNC machine's axis drives in combination with the EDM unit's actuator.

The contents/structure of the dissertation is outlined below.

Chapter 1: Introduction

Provides the context and purpose of the research and introduces the concepts of RMS and EDM.

Chapter 2: Background Information, Literature Survey, and Research Objectives

The EDM process and typical equipment is described, and its relevance in the TDM and RMS environment is discussed; literature survey is embedded in these descriptions/discussions. The research proposal is

identified, and the outcomes achieved during the research project are indicated.

Chapter 3: Description & Operation of Research Platform, and Outline of Proposed Integration

This chapter describes the structure and functioning of the existing CNC RMT equipment and the purchased EDM equipment, and outlines the research objectives and planned implementation methods in terms of integrating EDM onto the RMT, in broad terms. In particular, the research objectives of integration using the RMT's CNC control interface and hardware, and of integration using an external controller and an independent actuator, are outlined.

Chapter 4: Integration Using EDM Machine Controller Directly

This chapter considers in detail the methods used to integrate EDM onto the RMT by making use of the EDM machine's controller and servo drive (with or without an additional servo amplifier) to directly actuate the RMT's servo motors. The purchased portable EDM unit's control system is described as a precursor, and its stability is analyzed for reference purposes.

Chapter 5: Integration Using Mach-3 and EDM Machine Servo Drive Voltage Signal

The method of integration whereby the EDM unit's control signal is used (indirectly) to govern the steps taken by the RMT's servo motors, by informing the RMT's CNC controller whether to provide advance or retract commands, is described in detail. Implementation is via the CNC's controller hardware and via its programming interface, and makes use of 'G-code' and 'Macro' programming, all described in detail. Performance of this implementation is evaluated, and the possibility of multi-axis implementation is discussed.

Chapter 6: Integration Using External Controller

This Chapter describes the implementation of integration of EDM onto the RMT by means of providing an external controller, i.e. outside of the

purchased portable EDM unit, combined with a high speed linear actuator. The controller's deployment in hardware by means of data-acquisition devices, and in software by means of LabVIEW dataflow programming language, is described in detail. A method for physically obtaining the condition (as a signal) at the spark gap between workpiece and electrode is implemented, as is a means of filtering the gap condition signal for use with the controller. Various LabVIEW programs for single axis control, and for multi-axis control, are provided and their operation described. The performance of the programs is also evaluated. Finally, the modelling of an EDM process in LabVIEW is presented; that is, a model for workpiece erosion rate is developed and included into a control program where the instantaneous actuator position, with appropriate signal noise added, acts as a surrogate for spark gap condition, for control parameter tuning purposes.

Chapter 7: Modelling of EDM Control System in Scilab

In this chapter Scilab, which is similar to Matlab, is used to simulate the actuator, the controller (and amplifier), and even the EDM erosion process itself. Modeling is by means of Laplace Transform block diagrams. The method used to find the linear actuator's inherent parameter values, so that the actuator block diagram model could be made more accurate, is also described.

Chapter 8: Discussion and Conclusions

The outcomes of the project, in terms of the various developed methods of integration of EDM onto the RMT, are summarized and the performance of the different methods compared. The main objectives achieved during the research are clarified, suggestions for improvements are made, and areas for potential future research work are identified.

2. Background Information, Literature Survey, and Research Objectives

2.1 Overview of EDM and Description of EDM Process

[This section is in part a re-production of part of a document co-authored by the author of this dissertation, and others at NMMU [1]; the part being referred to was written by the present author, whereas the other authors focused on other aspects.]

EDM is a non-contact manufacturing technology, and was one of the first used – and remains the most common – 'non-conventional' machining process [8-10]. In an EDM process, or spark erosion process as it is sometimes called, high frequency electric pulses are produced between the tool electrode (usually the cathode) and the workpiece (usually the anode), by a spark generator. A dielectric fluid, present in the tiny gap between the tool and work piece, breaks down during the pulse and transmits the energy via a narrow plasma channel that forms. Erosion of the workpiece is in the form of tiny craters which appear on the surface, generally attributed to melting or vaporization of the workpiece material by thermal energy resulting from the electrical pulse. The dielectric also serves to flush away tiny pieces of debris that accumulate between the tool and the workpiece [8-16]. Figure 2.1 below depicts a (very simplified) basic arrangement of an EDM system, Figure 2.2 overleaf depicts the process occurring at the spark gap, and Figure 2.3 overleaf shows actual voltages recorded during a typical EDM process.

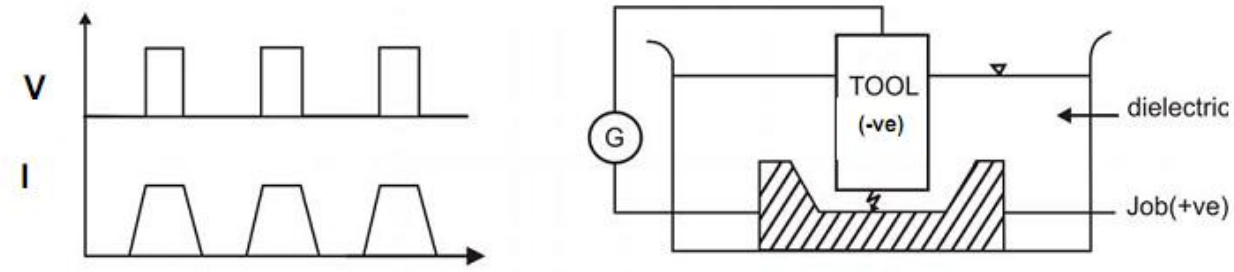


Figure 2.1 Basic Arrangement of EDM System [11]

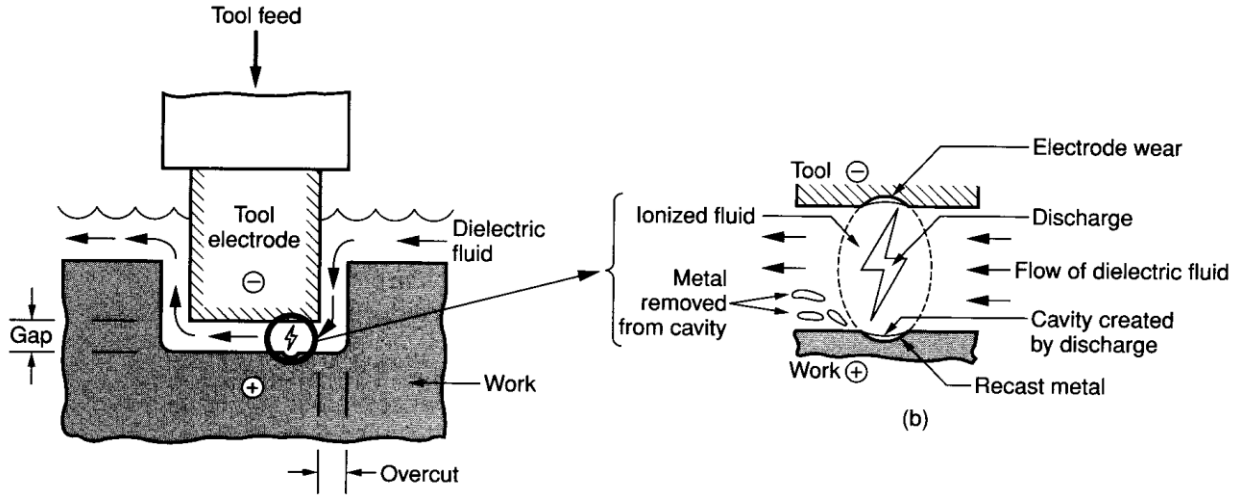


Figure 2.2 Process Occurring at the Spark Gap [10]

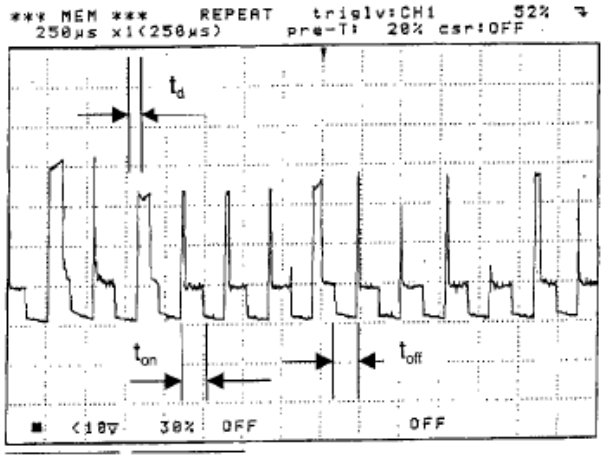


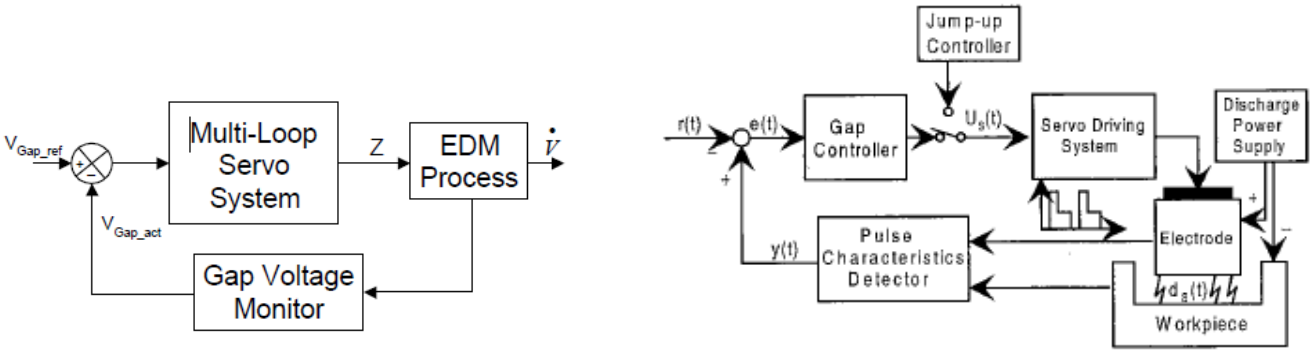
Figure 2.3 Actual Voltage Pulse Pattern Occurring Across the Spark Gap [16]

In the process, the tool (electrode) is also eroded to some degree, although at a slower rate than the workpiece [8-10]**.** Modern EDM machines are invariably CNC (Computer Numeric Control) based. The most common (and least expensive) EDM machines operate in a single vertical (Z) axis ram stroke (the workpiece is positioned manually using the X and Y axes). The tool electrode is slowly lowered into the work piece, eroding it as it travels, hence the name Sinking EDM or Plunge EDM. In the process, the workpiece gradually assumes the shape of the electrode, which is made of a relatively soft and inexpensive material and can be machined by conventional means (thus can be of a complex shape).

Process control is extremely important in EDM, especially to maintain the optimum size gap between the tool and workpiece. The process control methods are varied and are

vendor specific [8,9]. Thus EDM equipment usually is supplied as a complete unit comprising pulse generator, tool holder, vertical ram with servo motor drive, integrated CN controls with Human-Machine-Interface (HMI), and in most cases a tank, pump and filter system for the dielectric fluid. Various safety features (controls) are also common, e.g. to ensure that the erosion region remains submerged in the dielectric at all times.

Figure 2.4 below shows a typical control block diagram of the EDM process, where (b) includes the role of the pulse generator in the system, and shows a possible method for detecting the gap condition. Figure 2.5 depicts a typical servo motor drive as applied to EDM.



(a) [12] (b) [16]

Figure 2.4 Block Diagram of EDM Control Process

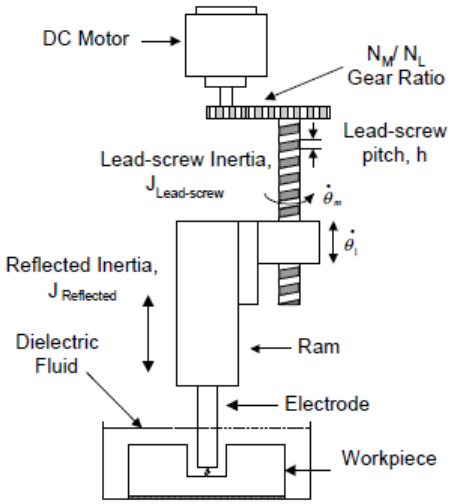


Figure 2.5 Typical Servo Drive for EDM [12]

The main function in the control loop of an EDM machine is to maintain desired conditions at the spark gap, i.e. the small distance between the electrode and the workpiece, which it does via a servo motor and lead-screw arrangement. Typically it is

desired that a gap on the order of 20 to 100 microns or so be maintained. Too large a distance and sparking does not take place (open-circuit), and too close, short-circuits occur. Also, surface finish is dependent on the gap distance, as is material removal rate. The control system constantly monitors the gap conditions and takes action by advancing or retracting the electrode via a servo mechanism. This adjustment occurs quickly, so that the electrode is effectively oscillating, but naturally the amount of advancement slightly exceeds amount of retraction, as the workpiece is slowly eroded. Different approaches to gap control exist, but a fairly simple one is the measurement of the average voltage across the gap (i.e. between electrode and workpiece), and the adjustment of the servo motor accordingly to maintain a constant (setpoint) voltage. This is because the voltage measured across the gap, and the gap distance, are closely related. The larger the gap, the higher the voltage will be across the gap (if sparking is occurring), as a higher voltage has to be reached before the dielectric breakdown voltage is reached, and this voltage is distance related. Another approach utilizes 'ignition delay' detection [13], (or On-delay detection) where the time taken for an arc to form is measured. Again, the larger the delay between the application of voltage to the gap and the actual striking of the arc, the larger the gap is. These relationships are not necessarily linear, but can form the basis of a control system. Other approaches include 'threshold' detection [14], and 'pulse characteristics' discernment.

Control schemes include PID (Proportional, Integral, Derivative), Adaptive Control, and Fuzzy Logic, among others [9,12-14,16]. In all control methods, an operator chosen setpoint (essentially the gap distance by inference) is specified, and the system attempts to maintain that setpoint by constantly advancing or retracting the electrode in tiny increments. The setpoint (together with other chosen parameters such as discharge current and pulse generator voltage setting) dictates such factors as material removal rate and surface finish.

Most commercially available EDM machines are dedicated machines, i.e. they are for the single purpose of EDM processing. Figure 2.6 overleaf shows a typical EDM machine.

Figure 2.6 Typical EDM Machine [17]

Machined parts must be loaded from one machine to another, if different operations are required on the same piece. The machines commonly have X and Y axis slides, but these are for initial positioning of the workpiece or electrode only, by manual means. More expensive machines may have three (or more) axis CN control. This is advantageous in that, for example, complex contoured shapes (common in dies and molds) can be machined from simply shaped electrodes e.g. square or round [18-20]; also, maintaining relative motion between tool and work piece (e.g. rotating the electrode and/or orbiting the electrode) can improve surface finish [8,9,21].

2.2 EDM in the TDM and RMS Environment

[This section is also in part extracted from the document referred to above [1], and was also written by the present author.]

Reconfigurable manufacturing systems are characterized by modularity, integrability, customization, convertibility and diagnosability [4-6]. Mehrabi *et al*. [22] defines RMS as “a system designed for rapid adjustment of production capacity and functionality in response to new circumstances by rearrangement or change of its components”. RMS allow for rapid changeover between products, rapid introduction of new products and unattended operation. A CNC based reconfigurable machine tool (RMT) can form the heart of an RMS [3,23].

EDM, alongside metal cutting (in particular HSM), is a major fabrication method used in the TDM industry [24,25], especially when hard materials are to be machined (e.g. tool steels, after hardening heat treatment, to avoid deformation), and when intricate shapes with sharp internal characteristics e.g. corners, thin slots, are required [8,9,24]. EDM is most suited, and often necessary, for the manufacture of the non-standard, functional portion in TDM, e.g. for the production of mould cavities and dies, of various free-form contoured shapes, specific to the part requiring manufacture. With the advent of new high speed cutting tools designed to machine even hardened materials, the need for EDM may well decrease. However, EDM remains an indispensable technology in the TDM industry [24]. I.e., EDM often presents the only feasible or cost effective solution for the production of some mould or die parts. Recent advances in the control of EDM processes have provided improved precision and surface finish qualities, as well as higher material removal rates. EDM is now tightly integrated into the TDM making process, and machining is often unattended [23,24].

Electrode (tool) wear can be minimized, but is generally present in EDM. Rajurkar and Yu have developed the 'uniform wear method' to predict the degree of tool wear, and integrated it into the EDM process to improve machining accuracy [24]. Other methods of tool wear compensation also exist [9,18,19,21].

The use of EDM in micro-machining is also common, for example, for the manufacturing of tiny moulds. EDM has many benefits in the mircro-manufacturing field. For example, fine deep holes can be produced due to the non-contact properties of the process [9,21].

Multi-purpose or flexible manufacturing systems may have EDM functionality. But in this case the EDM equipment is built into the system, and the equipment is specifically designed for this purpose.

In terms of mechanical aspects, machine tools are comprised of two broad categories of components: 1) structural and motion producing components, and 2) machining elements (tools). Thus, it is possible to create reconfigurable machine tools in a modular fashion. Structural elements can be removed, added, re-arranged, replaced by other modules with slightly different functions, etc. Likewise machine tools can be of a modular nature, and can be added, re-arranged, etc. An EDM tool (electrode) can be considered similar to a conventional machine tool in that it could, for example, replace a cutting tool,

for certain production operations. I.e., if the EDM tool comprises a spark generator and electrode only (no slides of its own), the electrode could be positioned in place of, for example, a spindle. This is possible as long as the CNC machine itself is of an open nature, i.e. designed to be able to potentially accommodate such additional and varied equipment, and, in particular, has open architecture controls. Several such machines have been developed [26]. An example is a Generic Modular Machining Platform, developed at Brunel University [23]. Intended for micro-machining, this system has been equipped with an EDM module. The EDM module is necessarily separate to the machine's motion axes. EDM was chosen to be integrated into the machine, to demonstrate its modular, reconfigurable nature, and because EDM is a particularly useful process in micro-manufacturing. A spark generator was purchased from a vendor, and integrated via the platform's open architecture control structure. A number of other researchers have developed EDM equipped machine tools using generators from EDM suppliers, and integrated them onto CNC based multi-axis machine tools [26]. In this way, the users can have access to the controls of the EDM process, in order to modify them – e.g. incorporate adaptive control, tool wear compensation, etc. – which may not be possible if a complete proprietary machine is purchased.

2.3 Research Proposal

[This section is also extracted from the document referred to above [1], and was also written by the present author.]

In order to further demonstrate the reconfigurability of the (milling) CNC machine tool developed at NMMU, it was proposed that an EDM tool be integrated onto the machine. An inexpensive single axis ram portable unit was to be purchased and combined into the current machine such that full 3 axis EDM can be realized. Reconfigurability would be demonstrated in that, although the machine is not designed specifically with EDM processing in mind, a commercially available self-contained unit (module) could be integrated into the system, both physically and in terms of control. The current open architecture CN control would need to accommodate the EDM unit, in that it is essential that the movements of the machine's axes and the sparking process be very finely coordinated. The EDM unit could maintain some of its internal control function, but outputs extracted from it would be utilized by the main CNC program. The EDM unit purchased would be much simpler (hence far less expensive) than a normal EDM

machine, in that it would have no other slides apart from the driven vertical axis (ram), and no machine base. The ram axis travel could also be very small without comprising the system's functionality, as all travel could be executed by the current reconfigurable machine's own axes, or alternatively, the RMT's Z-axis travel could complement (extend) the EDM module's travel.

2.4 Outcomes Achieved in the Research Project

Some of the above objectives were achieved as a result of the research, or at least partly achieved, but scope for further improvement exists. During the project, it became clear that the incorporation of the direct control of the movement of the electrode onto the control system of the CNC would be beneficial, and this became a focus of the research. Integrating the EDM process control into the overall movements of the CNC axes has to a large degree been achieved in terms of single axis machining; in terms of multi-axis machining, combining of the CNC's path generation function and the EDM control has not been achieved as originally envisaged; however a version of multi-axis machining has been developed, as will be explained later. In addition, a PC-based simulation model of the control of an EDM electrode servo movement has been developed, which could be used for optimizing various control parameters. Also, in addition to purchasing the low cost portable EDM tool, a linear actuator was included onto the CNC tool, such that more effective multi-axis control could be achieved.

2.5 Conclusion to Chapter 2

The EDM process has been described, and the relevance of EDM in the TDM industry has been identified. In the light of these, a research proposal has been developed, and in addition, the outcomes achieved have been summarized. In Chapter 3 following, the research platform is described, and its operation explained in more detail, so that an outline of the proposed methods of integration of EDM onto the RMT can be provided.

3. Description & Operation of Research Platform, and Outline of Proposed Integration

3.1 Portable EDM Machine for Integration

Portable, inexpensive EDM tools are not common. A suitable machine, namely the Alic-1 produced by Sure-First of Taiwan, was however sourced and purchased. The tool comprises the following, as can be seen in Figure 3.1 below.

a) The 'spark generator' or power-pack module
b) The electrode servo-mechanism (actuator), sometimes also referred to as the servo head or electrode head
c) A tank and pump for the dielectric fluid

(a) (b)

Figure 3.1 (a) Servo head of Alic-1 EDM Machine and (b) Complete Unit Comprising Power-Pack and Dielectric Fluid Tank, with Servo Head shown mounted on a CNC Machine [27]

The EDM tool's control function is essentially embedded in the power-pack module (at least physically), i.e. it is not a visibly separate module; and it was not known initially how much overlap there was between the control function and the spark generation function. The control function is primarily to maintain the electrode at a constant, ideal distance above the work piece, such that optimal sparking can be achieved. This distance, among other parameters, can be adjusted according to the desired machining

characteristics. The Alic-1 front panel of the power-pack is shown in Figure 3.2 below; various adjustment dials are visible. The dielectric fluid pump and tank are shown in Figure 3.3. EDM is most often performed with the workpiece submerged in a bath of the dielectric fluid, but in the case of the Alic-1, a nozzle is used to guide fluid over the region where EDM is taking place. Thus just a small fluid capture tray was needed (under and around the T-slot machine table), to collect the fluid and route it back to the tank.

Figure 3.2 Front Panel of Alic-1 Power-Pack/Controller

Figure 3.3 Dielectric Fluid Tank and Pump of Alic-1

The research ideas and objectives were generated with the decision to purchase the Alic-1 tool in mind, and in general were a natural process of evolution as certain knowledge was accumulated, i.e. the realization of the shortcoming of one aspect lead to the development of another.

3.2 Portable EDM Machine and Integration with Computer-Numeric Controller

[Refer to section 3.2.1 for further description of the RMT and its CN controller]

Typically the servo-head of the portable Alic-1 machine is mounted on a CNC machine, simply for positioning purposes, and such that the part being EDM'ed can be secured. However, the EDM machine remains completely independent of the CNC machine in terms of control. An initial idea was naturally simply to mount the Alic-1 servo head on the CNC machine, and investigate the level of integration that could be achieved in terms of communication between the CN controller and the EDM machine, in terms of the CN controller giving simple commands to the Alic-1, such as starting EDM at a particular point during the execution of a CNC program. The servo head mounted on the RMT can be seen in Figure 3.4 below, positioned in front of the spindle.

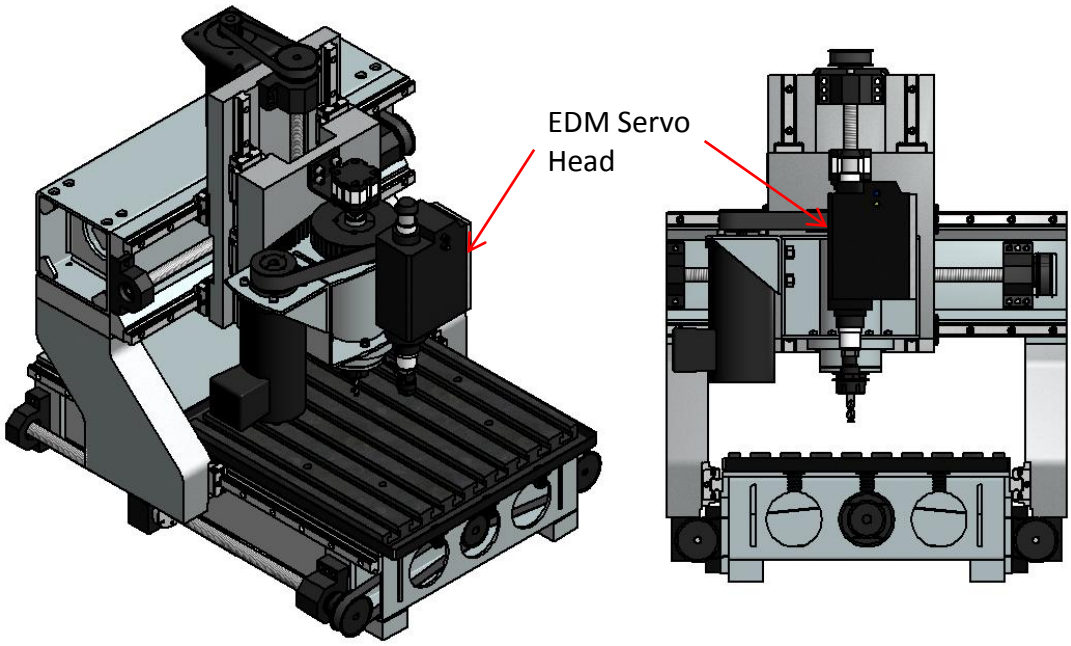


Figure 3.4 EDM Servo Head mounted on RMT [2]

For co-ordination between the CNC RMT and the EDM servo head, it is important to know the offset distance between the electrode position, and the machine spindle, in the X and Y dimensions. The reason being, that sometimes one wants to perform a milling operation on a workpiece, and then, without removing it from the machine, i.e. without resetting, one wants to use EDM on the workpiece, perhaps for achieving a certain surface finish, for example. For this purpose, among other reasons, an inexpensive webcam was purchased. The RMT's CN controller program (Mach-3 from Artsoft – see

section 3.2.1) has certain ‘add-on’ capabilities. One of them is for the use of a webcam. The webcam was mounted under the spindle housing, viewing directly downwards (towards the workpiece). On the CNC PC, the picture can be viewed, and the add-on creates a thin circle, centered on the center of the view range of the webcam. The cam can be focused onto the surface of the workpiece. By drilling a blind hole using a cutter, and then performing EDM with a round electrode of similar size as the cutter (without moving the workpiece in-between), the distance between the drilled hole and the EDM hole could be measured, using the webcam (the CNC axes’ are jogged until the view of the drilled hole, and then the EDM’ed hole, fits concentrically to the thin circle on the PC screen, and the co-ordinates at those positions are noted on the CN controller screen). In this way it would be possible to specify tool offsets in the CN controller setup, such that cross-over from milling to EDM could be achieved simply and accurately.

The main initial research objective was to achieve a more integrated system, by using the CNC machine’s path generation function (i.e. in Mach-3) in conjunction with the EDM unit’s own controller, to achieve single and multi-axis machining, by utilizing the servo-drives of the CNC tool’s own linear movement axes. In terms of control, this arrangement would be as envisaged in Figures 3.5 below and 3.6 overleaf, where one can see the relation between the positioning commands given by the CN controller (Mach-3), and the EDM unit’s control function. Here it is assumed that the Alic-1 EDM machine utilizes some form of gap voltage detection as the basis of its control. The reader is again referred to section 3.2.1 below, in order to better understand the figures. ‘Mach-3 Main’ and ‘Mach-3 Macro’ refer to parts of the CN controller program (so-called Macro’s are essentially sub-routines which are called by the main program). ‘Gecko Drive’ in these figures is a driver for a servo-motor of the CNC RMT.

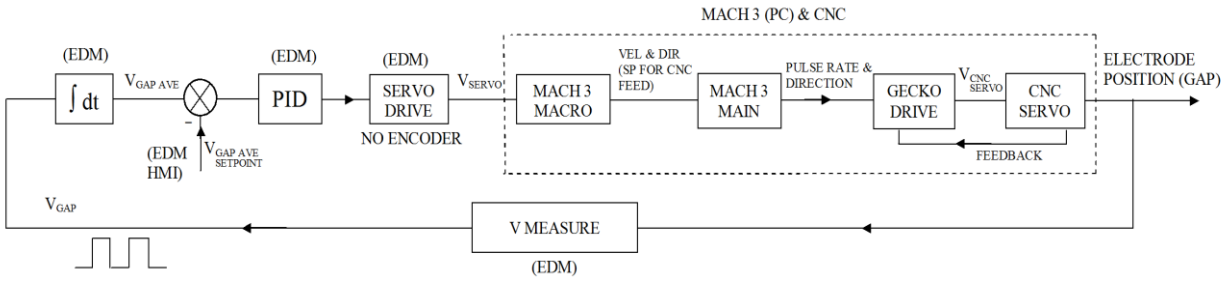


Figure 3.5 Envisaged Integration of EDM Machine Control and CN Control of RMT (Single Axis EDM)

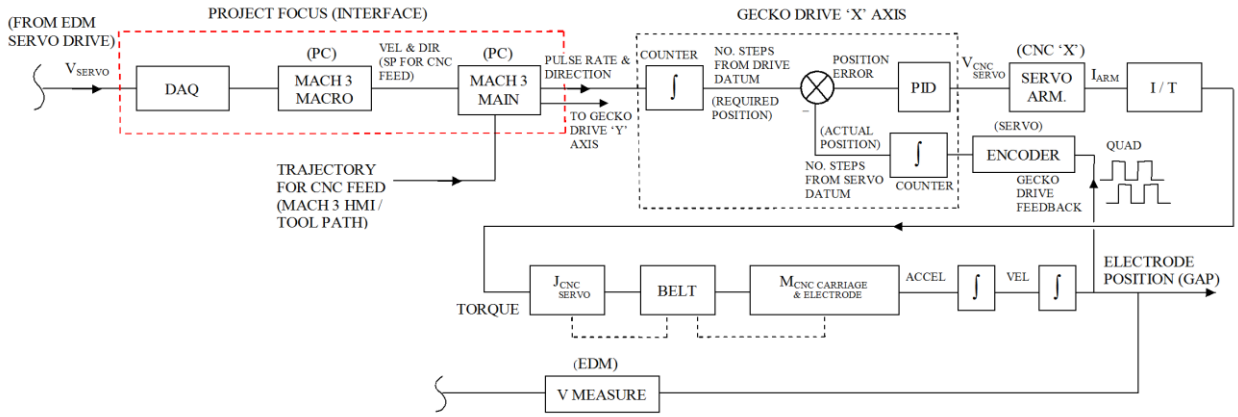


Figure 3.6 Envisaged Integration of EDM Machine Control and CN Control of RMT (Multi-Axis EDM)

A concern always existed that the movement of the CNC tool's axes may not be fine enough, in terms of resolution, or may not be stable enough (e.g. due to higher inertia relative to the EDM tool's own servo-drive mechanism), thus the potential inclusion of an additional actuator(s), was borne in mind.

An additional concern was the feasibility of the use of the CNC tool's path generation software and hardware, with the EDM process control. It was found that utilizing the CNC's movement commands was not simple in the case of multi-axis movement, and control stability would be even more difficult to achieve in a multi-axis environment.

Thus after some initial testing, a decision was made to purchase an additional actuator, for fine control. Also it was decided that the development of a control system essentially separate to the EDM machine or the CNC tool would be beneficial (this system could however be made to communicate with the RTM's CN controller, thus achieving one of the original objectives of the project; also, if PC-based, it could potentially run on the same PC as the CN controller).

3.2.1 Overview of CNC Machine Tool System

The CNC RMT onto which the EDM tool was to be integrated is fundamentally a gantry type three axis milling machine incorporating servo motors driving ball-screws, via timing belts. Figure 3.7 below is a repeat of Figure 1.2 above, shown here due to relevance. Figure 3.8 shows part of the control panel.

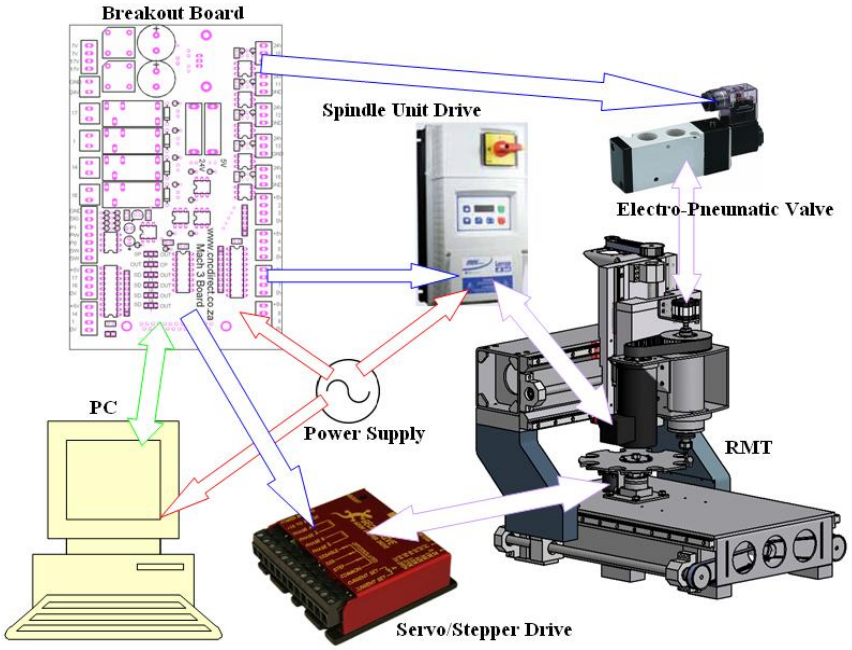


Figure 3.7 Reconfigurable Machine Tool, showing Tool and Control System Overview [2]

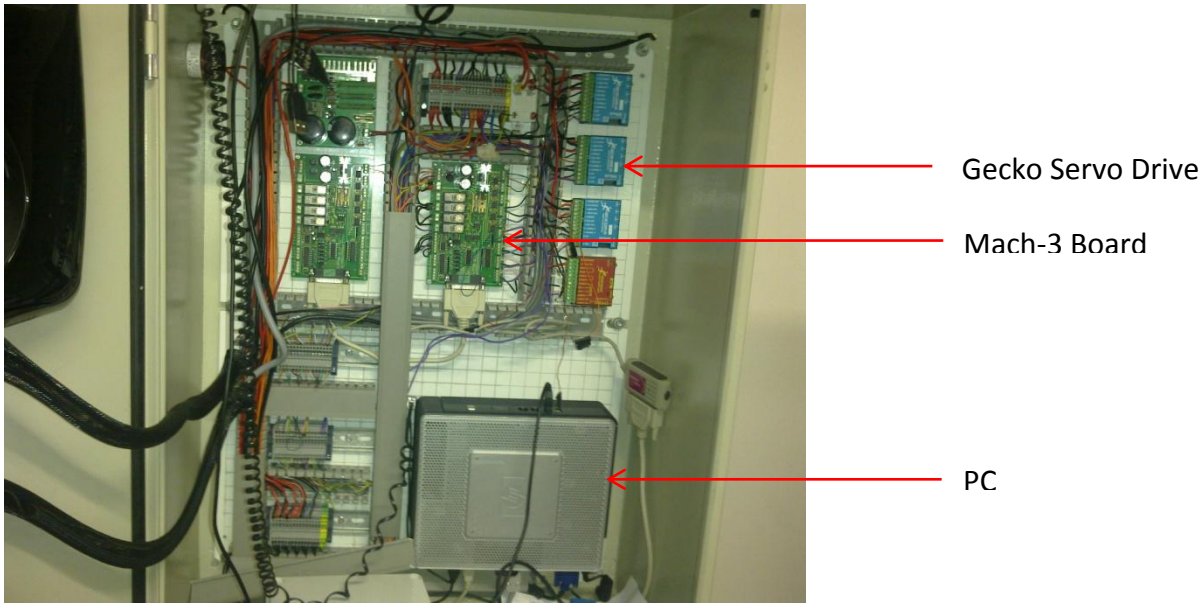


Figure 3.8 Reconfigurable Machine Tool Top Panel: Mach-3 Boards, PC, and Gecko Servo Drives

The machine is controlled using Mach-3 software from Art-Soft. A free version of Mach-3 is available but the university has a license for the full version which has higher functionality. The control panel houses a PC and screen, power supplies, the Mach-3 breakout board, the three (DC) servo-motor drives (X,Y,Z axes), variable speed drives for a conventional and high speed spindle, and some relays. The Mach-3 CNC control program runs on the PC and has the function of providing positioning (and other)

commands to the machine tool, as well as providing a HMI for the user. The software communicates to the machine via the PC's parallel port, which is connected to the breakout board (Figure 3.9).

Figure 3.9 Mach-3 Breakout Board [28]

The breakout board has digital I/O, and an analogue output. Some of the digital I/O are used for e.g. limit switches and emergency stop, and some are for sending pulses (position commands) to the servo motors' drives (Gecko G340 servo-drive). A typical Gecko servo-drive is shown in Figure 3.10 below.

Figure 3.10 Typical Gecko Servo Drive [29]

The drives (one for each axis) receive these pulses and encoder signals from the motors themselves, and produce a pulse width modulated (PWM) driving voltage to the servo motors' armatures to effect axis movement to the commanded positions. The Mach-3 software utilizes so-called 'G-Code' programming to specify X,Y,Z position commands, and for spindle speed control and tool-changer control. A short summary of G-Codes can be found in Appendix A. So-called 'Macro's' can also be written for Mach-3, which are essentially subroutines called by G-Code programs. Various variables e.g. the value of digital readouts, and the logic state of input and output pins and LED's on the HMI screen, are accessible to the macro's, which utilize VB-Scripting language.

Position commands are given to the drives of the servo motors by means of digital ‘step’ pulses, via the Mach-3 breakout board. A ‘direction’ input is also held high or low while each step pulse is received, indicating advance or retract of the relevant axis. This aspect of the control system is open-loop in the sense that the CNC program is not aware whether the commands have been successfully obeyed by the axis servo motors (a fault signal is however produced by the Gecko drive if the motor’s movement is not able to follow the commanded position, i.e. if its position error exceeds a certain value). The Gecko drives receive the digital position pulses, and increment or decrement counters in the drives with each pulse according to the state of the direction input. Meanwhile the Gecko drives also receive quadrature encoder signals from the servo motors, which are duly interpreted as physical steps being taken by the motors, and the corresponding direction. In a similar fashion to the signals received from the breakout board, counters are incremented or decremented accordingly. Thus each motor drive keeps track of a commanded position, and the actual position of a given motor. The count difference between the counters holding these positions is a measure of the position error of the motors. This difference is amplified via a PID control loop to provide a varying voltage (i.e. pulse width modulated) to the motor armature via an H-bridge which allows for bi-directional control of the motor. The Gecko drive’s control loop constantly tends to minimize the position error by altering the voltage signal to the armature to drive the position error towards zero. A block diagram of the Gecko drive function is shown in Figure 3.11 overleaf [30].

The Mach-3 program includes linear (and circular) interpolation such that, in the G-Code, only the final co-ordinates of a desired position to be achieved by the end of a move command need be specified (in addition to a feed speed at which the move must be made). The software is capable of handling absolute or incremental type position commands. The step pulses for each axis are produced in the correct ratio for the three axes, and at the correct frequencies such that the velocity as well as position of each axis is controlled during the move. Although the commands to the Gecko drives are ‘step- and direction’ pulses, motion of the motors is usually smooth, not in individual steps, as the pulses are received usually at fairly high frequencies, to effect the desired feed rates; the PID control will respond to an individual step, but most likely while the motors are in motion, the position error would be a number of steps, i.e. the steps are just a digitized way of measuring the position error (as opposed to for example using a tachometer (for velocity error detection), which is an analogue device).

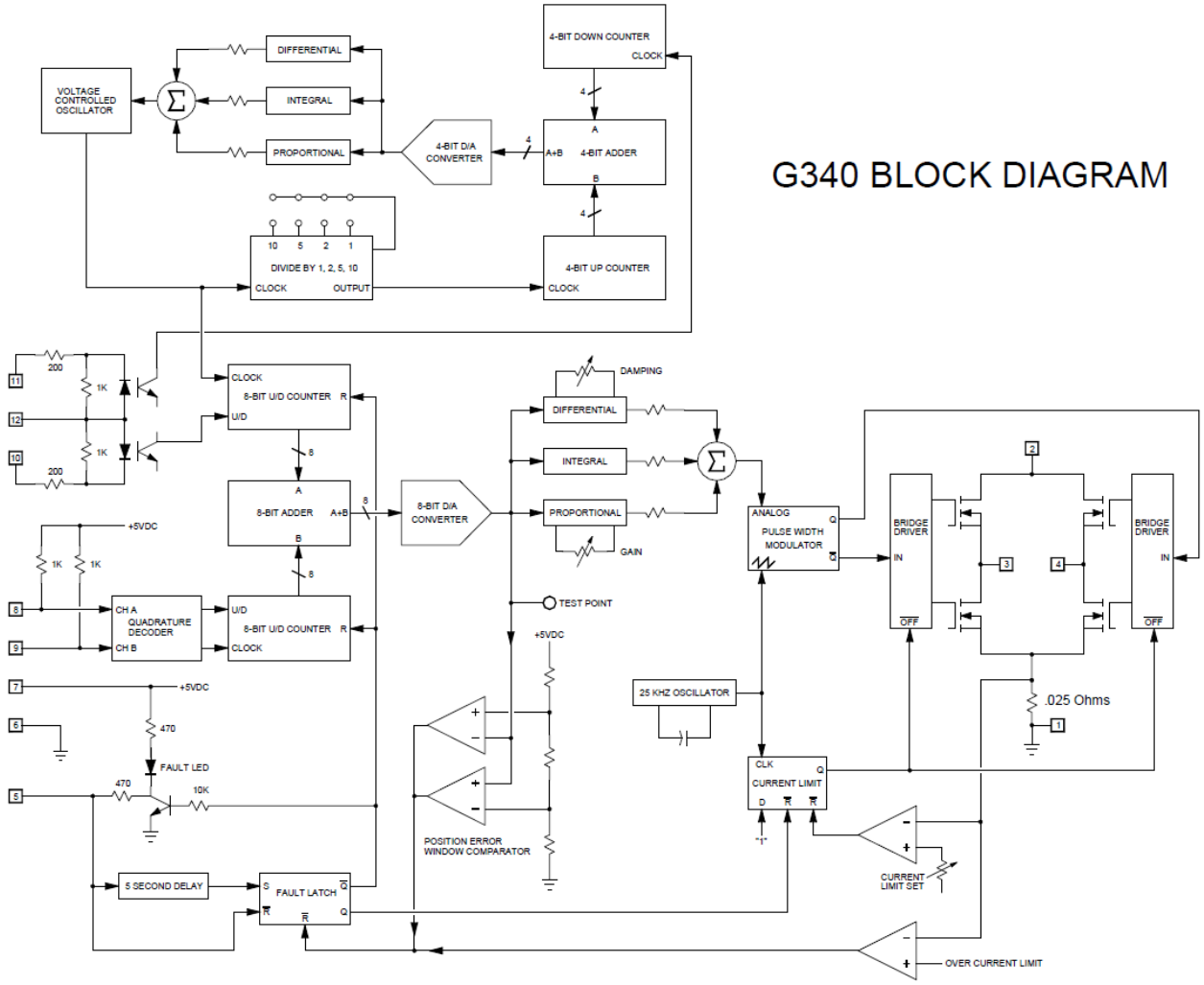


Figure 3.11 Gecko Servo Drive Control Block Diagram [30]

In terms of the objectives of the project, a main aim is to combine the use of the Mach-3 software and G-Code programming, with the control signal from the EDM tool. Thus Mach-3 would dictate the relative movements of the axes, while the EDM tool's control system would modulate the execution of those movements in accordance with the conditions at the spark gap. To understand this dependence between the two systems better, further clarification of the EDM control process is required.

3.2.2 Further Discussion of EDM Machine and Mach-3 Integration

In section 2.1 the EDM process and control philosophies were described. It was stated there that the main function of the control loop in an EDM machine is to maintain desired gap conditions (essentially distance), which is achieved by movement of the electrode servo motor. Because of this, it is clear that it is not possible simply to provide a fixed feed rate to move the electrode to achieve EDM, as if just a fraction too fast, the

electrode will approach the workpiece too closely, short circuiting will occur, and no erosion of the workpiece will occur; meanwhile the electrode would continue to advance, and collide with the workpiece. If the feed rate was of course set extremely slow, EDM would be possible, but could never be at a high material removal rate, as one would not know exactly what the maximum allowable feed rate is, for a given arrangement – this is because the advance rate is dependent on a number of factors, including electrode material, workpiece material, gap voltage setting, current setting, on-time setting, off-time setting, electrode cross-section area, gap flushing conditions, and the specifics of the a particular EDM machine's operation (there are subtle, and sometimes not-so-subtle, differences between individual machines, not to mention various manufacturer's machines). In addition, even if a very slow feed rate were set, the random production of debris in the gap could produce a gap that is effectively too small, in which case short-circuiting would occur, and no erosion would take place (even though the electrode would try to advance – ensuring that more short-circuiting will occur and thus no erosion).

Thus, although G-Code can specify the relative feed rates (ratio of pulses) for multi-axis EDM, the execution of these pulses must be somehow tied to the gap conditions, i.e. if gap is too large, advance through the pulses, and if too small, retract according to the same pulse ratio. And not just 'advance' and 'retract' commands are required, ideally the speed of advance and retract (at a given axis step ratio) is also required, as the larger the gap is relative to the optimum, the faster one wants to correct it towards the optimum sparking distance by advancing, and likewise the smaller the gap relative to the optimum, the faster one wants to retract (so that as much time is spent under the correct sparking conditions as possible). Even if single axis EDM is being considered, from an initial position to a specified final position (depth), one cannot force a feed rate but must let the gap conditions dictate the feed.

3.3 Combining External Controller with EDM Machine

When using an independent EDM machine, access to its control signals is potentially not possible, or at least difficult. Also the way that the machine would communicate with servo motors of a CNC machine would not necessarily be ideal. Thus it may be preferential to obtain a more direct signal from the EDM process itself, and use this with an independent controller. The independent controller can be specifically suited to the

CNC machine's servo motors, or whatever actuator is being used. This would be especially true if multi-axis EDM is desired, where axes' travel must be co-ordinated. Much of the project focuses on this research area, but with the idea of utilizing an independent, small actuator, potentially working in conjunction with the CNC machine's servo motors.

3.4 Conclusion to Chapter 3

In this chapter the research platform and its operation have been described in detail, and the proposed methods, or rather broad categories of methods, of integration of EDM onto the RMT have been outlined. In Chapters 4 to 6 following, each of the main integration categories proposed is focused on in turn. The methods of integration are depicted in control block diagrams and described in detail. Results obtained from each method are displayed where possible, and discussed. Attention now turns to the first category, namely integration using the EDM machine's controller directly, in Chapter 4.

4. Integration using EDM Machine Controller Directly

4.1 Alic-1 Control System

The Alic-1 employs a (DC) servo motor to effect movement of its (vertical, Z) axis. The motor has a position encoder (presumably quadrature). However, it appears that the encoder of the motor is not actually used during the EDM process itself (the encoder's purpose is explained later), i.e. no position commands are specified by the control system for the purpose of maintaining desired gap conditions. Instead, a varying voltage is simply applied to advance and retract the electrode, based on the voltage across the gap, i.e. the control variable. A basic control function block for this system is shown in Figure 4.1 below. (Note: it is not certain whether the Alic-1 employs a back-emf based inner control loop – in which case the diagram below is an oversimplification.) Again it is assumed that the Alic-1 employs average gap voltage detection as its means of gap condition regulation.

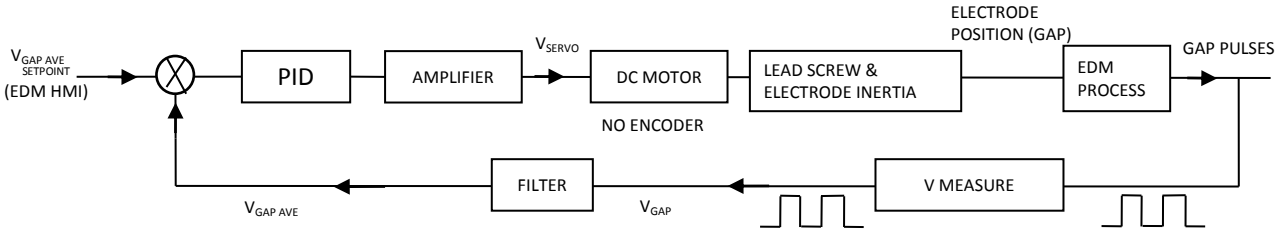


Figure 4.1 Basic EDM Control System (Alic-1)

The control system does not specifically need the position encoders (although there are good uses for such in EDM process control), as inherently when the electrode advances too close to the workpiece, the gap voltage drops, and when too far away, it rises. Thus actual position need not specifically be known. This is also a 'self-integrating system', in that the longer a specific gap voltage error exists, the larger the result of control action becomes, in a certain sense. This is because although armature voltage can be directly related to the gap error, this voltage essentially causes a velocity, and this velocity, when applied over a time period, produces an increasing position altering effect (and thus an increasing gap voltage altering effect). Furthermore, in the absence of friction, there need not be any steady-state error arising from this control, due to the self-integrating

effect. It appears the encoder is used only for display of the current position of the electrode in terms of travel from a datum position, and to stop the process once a specified distance of travel (i.e. sinking distance into the workpiece material) has been reached; also often a 'jump' function is used in EDM to help flush debris from the gap, by regularly quickly moving the electrode away from the workpiece for a short period, and then back, and it appears the encoder is used to force the electrode to a specified jump height and back to its position prior to the jump, after which the EDM process continues.

4.2 Measurement of Control Stability and Electrode Movement

A crucial investigation was to determine whether the CNC machine tool's servo motors could satisfactorily achieve stability of the gap conditions, i.e. a stable control loop. Control of the EDM process is very fine, with just a few microns separating the formation of a good arc, and a short- or open circuit. Fast response is also desired, in that gap conditions are constantly changing e.g. due to the continuous formation of debris particles in the gap. The EDM process is highly stochastic (i.e. probability-based) in nature, and control must adapt to the constantly changing gap environment.

There were essentially two concerns regarding the use of the CNC tool's axes for direct electrode movement, i.e. to control the gap, namely the much higher inertia of any of the axes' moving parts, and the presence of timing belts between the motors and the ball-screws, i.e. the elasticity, or compliance (or lack of stiffness) of the arrangement. The least affected axis is the Z (vertical) axis, as the Y axis carries the Z axis actuator (servo mechanism), and the X carries the Y axis actuator (and therefore the Z as well). Thus the Z axis was investigated since if it could not be satisfactorily used, then neither could the Y or X axes. It should be mentioned that one advantage of the use of timing belts over meshing gears is that they have essentially no back-lash, or play, when movement direction is changed.

In order to compare the performance of control via the CNC's motors vs. control by the EDM's own servo motor, it was necessary first to measure the range of motion of the EDM electrode when under its normal control, i.e. when being moved by the EDM's own servo motor, as a reference. This would also enable one to get an indication of the likelihood of successful control via the CNC motors. For this purpose, an LVDT (Linear

Variable Differential Transformer) position sensor was purchased from Applied Measurements Limited, UK. A typical LVDT position sensor is show in Figure 4.2 below.

Figure 4.2 LVDT Position Sensor [31]

The sensor has a measuring range (i.e. 'full-scale' travel of the shaft) of only 1mm. It was necessary to choose a sensor having such a small range, as the specified sensitivity and resolution of the devices was relative to the range, i.e. smaller range, smaller resolution. In the case of EDM, one needs to be able to detect position changes on the order of a few microns, hence the choice of a sensor with a small full-scale range. The LVDT chosen has a sensitivity of 10V per mm. An LVDT is a suitable type of sensor for achieving a very small resolution as it operates on the principle of the ratio of overlap of two current carrying coils (one stationary in the housing of the sensor, and one moving with the sensor's shaft), to induce a proportional voltage in the secondary coil where the primary coil is excited with a fixed, accurate, known AC voltage. Thus there are no real 'steps' in the measurement as any coil can be partially overlapping another, with no physical contact between them, and the resolution then is really only limited to the quantization error of the output (if digital), and/or stick/slip of the movement between the shaft and the housing. The output of the sensor was read into an analogue input channel of a data acquisition (DAQ) card (National Instruments ELVIS development board), and displayed in National Instruments LabVIEW using a standard virtual oscilloscope function designed for use with the ELVIS DAQ board. The spring return option was chosen for the LVDT shaft (or rather spring extension, not return, as the spring maintains the shaft extended), with ball end, such that movement can be detected simply by a surface pushing against the shaft end, i.e. no physical joint need be provided). Although this option made the mounting of the sensor simple, one concern was whether the spring would return the shaft fast enough to match the oscillatory motion of the electrode. The specifications indicated however that it would be sufficient,

i.e. stiff enough (with the shaft mass small enough), based on indications from a literature source that the oscillations of the EDM tool being investigated in that research was only on the order of 40 Hz [16] which is relatively slow; it was assumed that the machine now under investigation would not be very different (although the oscillation frequency is of course dependent on the mass of the electrode and its holder).

Figure 4.3 below shows the servo head with horizontal 'cantilever arm' attached at the top. The LVDT was mounted on the side face of the servo head, its shaft facing upwards and pushing against the underside of the cantilever arm. Care had to be taken to isolate the LVDT sensor from the EDM electrode (by applying a thin film to the underside of the cantilever), as the shaft visible in the picture is live, even at the top.

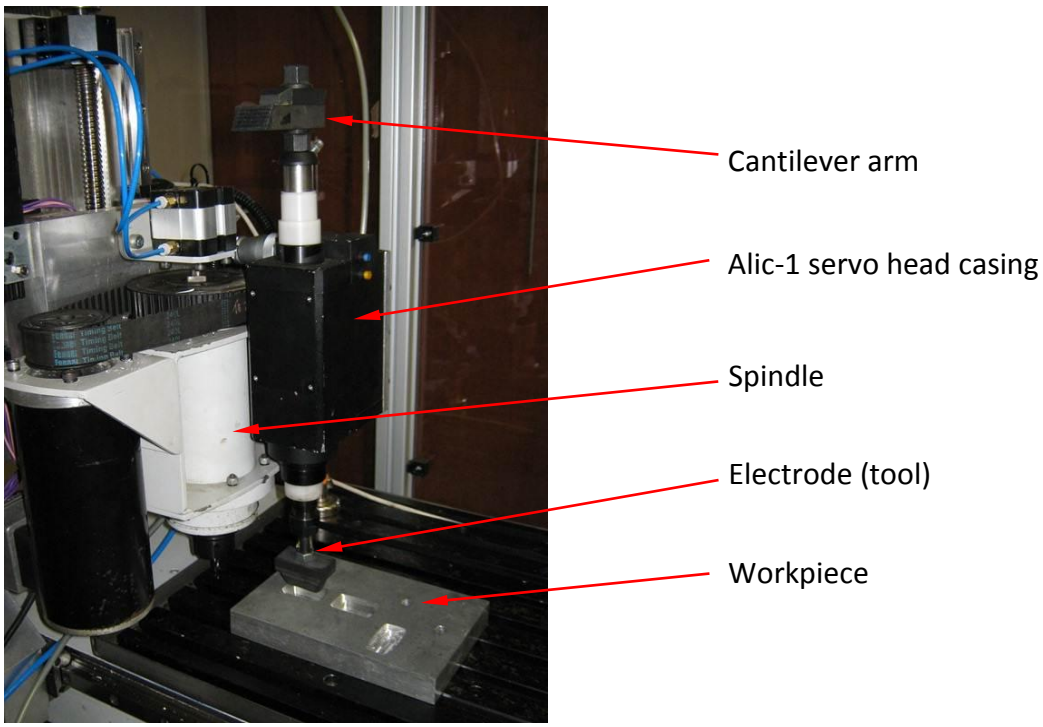


Figure 4.3 EDM Servo Head with Cantilever Attachment

The moving part of the EDM electrode linear guide rod extends right through the servo head housing, with the attachment point for the spark voltage supply cable being at the top. For measurement purposes it was easier to measure the relative motion of the electrode at the top i.e. above the servo head, by fixing the LVDT sensor to the head housing, facing upwards, with a horizontal extension arm being fixed to the vertical slide, pushing downwards on the sensor shaft as motion occurred. The sensor was mounted so that its body could be easily adjusted vertically, such that the electrode movement was within its small 1mm range (bearing in mind too that as erosion takes place, the

electrode movement soon shifts out of range and the sensor has to be adjusted again). Typical results of the motion of the electrode measured from these tests is shown in Figure 4.4 below, where the data captured by the ELVIS oscilloscope was imported into Excel for reproduction of (graphical) display. Since the sensitivity of the LVDT sensor is 10V per mm, the 0.4V approximate range of each oscillation of the voltage signal represents a movement range of about 40 micron (i.e. 20 micron either side of a midpoint). This is in agreement with [16] which shows (see Figure 4.5 below) a similar range of movement (electrode about 80 microns away from workpiece surface, with motion range approximately 15 microns on either side of the midpoint). The frequency of oscillation can be seen to be about 20 Hz, since there about 2 cycles of oscillation every 0.1 s.

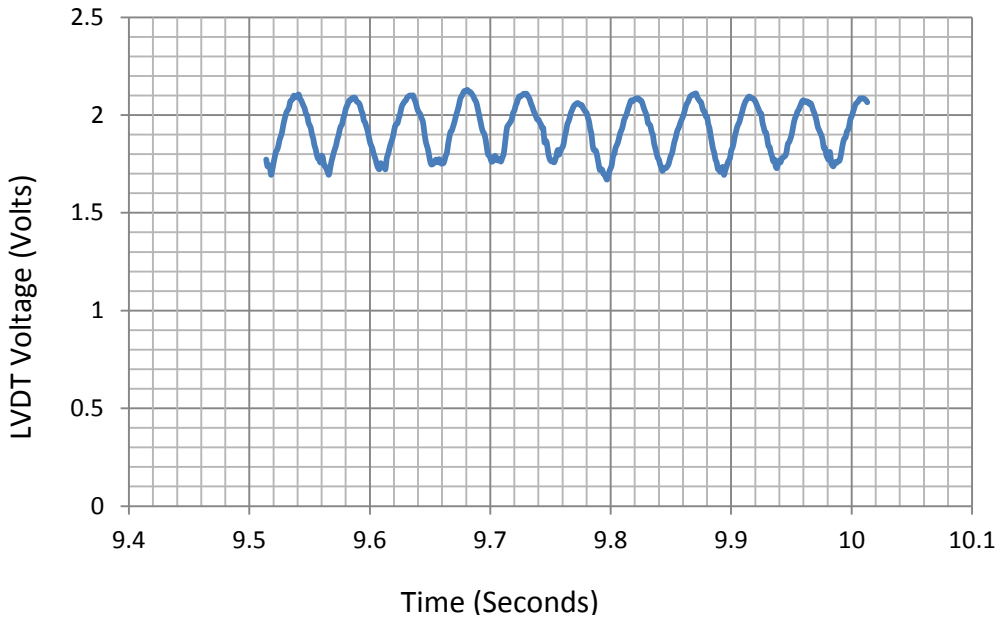


Figure 4.4 Typical Results of the Alic-1 Electrode Movement, Measured by LVDT

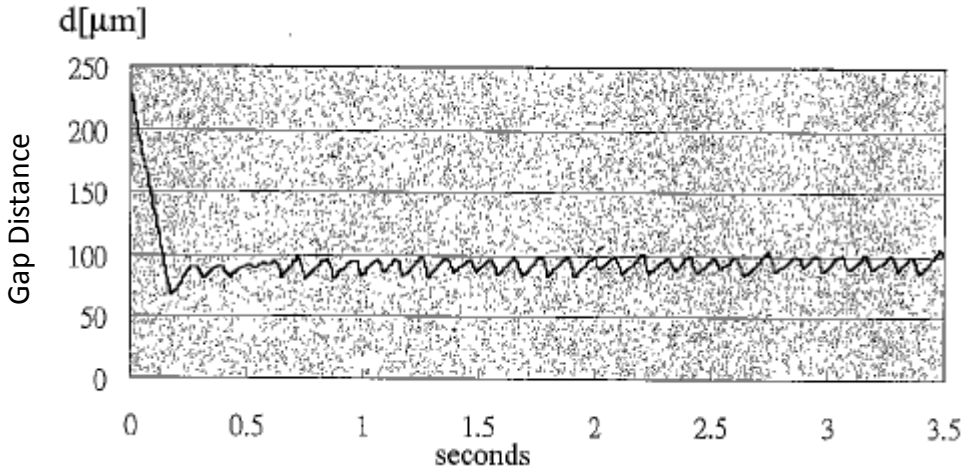


Figure 4.5 Gap Distance vs Time for Equipment of Reference [16]

The corresponding typical servo motor voltage of the Alic-1 is shown in Figure 4.6 below for comparison, to illustrate the relationship of the electrode movement and the servo motor voltage producing that movement (not captured at the same time as the LVDT graph). The measured frequency is indicated as 17.58 Hz by the oscilloscope, which is similar to the 20 Hz determined from the LVDT graph.

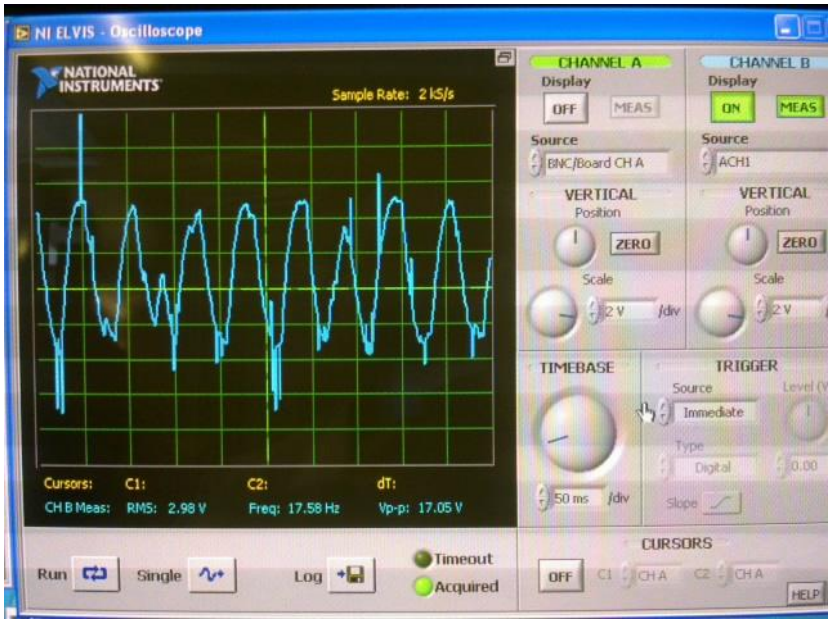


Figure 4.6 Corresponding EDM Machine Servo Motor Voltage

Since the material removal rate of an EDM process is very slow (overall advance rate of the electrode is on the order of 0.2 mm per minute, depending on the machine and settings), a good indication of the stability of control can be obtained from measuring the movement of the electrode relative to a fixed datum (as opposed to measuring relative to the surface of the workpiece, which is naturally more ideal, but more difficult to achieve). If measurements are taken over a relatively short period of time (say a few seconds), then the range of motion of the electrode seen relative to a fixed datum is almost the same as what would be seen relative to the workpiece. In fact, from a control point of view, [16] states that the workpiece surface erosion rate is so slow relative to the fast changes occurring in the gap due to debris production and removal and other stochastic processes, and relative to the instantaneous speed of the electrode (under-and over shooting), that it can be completely ignored in terms of evaluating the stability of a control loop, i.e. if the loop can control stably to a fixed position, it can control stably to the ‘moving’ (eroding) surface of the workpiece – bearing in mind of course that there is a lot of noise present in the control system, as will be discussed later.

In the method of control being considered, i.e. electrode motion being produced by the CNC machine's own axes, naturally the EDM tool's own servo drive would be immobilized, if the head (holding its electrode) was to remain on the CNC machine simply for mounting purposes, i.e. ease of 'attachment' of electrode to the Z axis travel. (This would of course provide additional mass (inertia) on the Z axis travel, which is not ideal for EDM control purposes; a lighter mounting arrangement could be considered, but relative to the inertia of the Z axis as is, the servo head would probably not cause a dramatic difference in performance.) In some instances, due to concerns regarding how Alic-1's control system would interpret a disconnected servo motor, the electrode was rather mounted by fixing it to the side of the servo head, and letting the EDM machine's servo motor 'go through the motions' while EDM was occurring using the CNC machine's servo motors, i.e. it was not disconnected, .and was free to move.

4.3 Using EDM Machine's Own Servo Amplifier

The simplest method for testing the Z axis of the CNC machine would be simply to 'hijack' the driving voltage of the EDM unit's own servo motor, i.e. to put the Z axis servo motor in the EDM unit's control loop, the power being provided by the EDM unit's own servo drive (amplifier). The servo drive is an analogue drive based on a power op-amp, the voltage supplied to the motor ranging from 0 to 10V (and 0 to -10V as it is a bipolar arrangement such that the motor's direction can be controlled). This is a much lower voltage than the rated voltage of the CNC machine's servo motors, meaning that the motors would not likely have much driving acceleration (as might be required by the control system). The CNC motors are rated up to 72V, but are being used with a 38V power supply on the CNC machine. A test was conducted to determine the current drawn by the EDM machine's servo motor during operation, by including a low resistance shunt resistor (of known resistance, namely 0.2 ohm) in series with the motor's armature, and measuring the voltage developed across it using an oscilloscope. 0.2 ohm was known to be an order of magnitude less than the resistance of the EDM machine's motor, and thus would not significantly affect the current draw. The maximum voltages measured were in the range of approximately 0.2 to 0.4V (depending on the gain settings on the EDM machine's front panel), meaning that the maximum current drawn was in the range of 1 to 2A, by a simple I = V/R calculation. This is significantly less than the rated stall (i.e. stationary armature) current (about 5A) of the CNC machine's servo-motors, thus it could not be guaranteed that the power op-amp of the EDM tool's drive

would handle the current that might be drawn by the CNC machine's servo motors, and damage might occur if the motor was driven directly from this power source. This is especially true since when a DC motor is forced to change direction suddenly, from full speed; up to twice the stall current can be drawn.

4.4 Using Independent Amplifier, with EDM Machine's Servo Drive Signal

The next simplest test was to use the actuating voltage being supplied to the EDM machine's motor as a control signal to force a correspondingly larger actuating voltage onto the CNC machine's Z axis servo motor. For this purpose an additional electrode head cable was purchased from Sure-First, such that, at the connection between it and the original cable (by means of terminal connectors as shown in Figure 4.7 below) the servo motor voltage could be 'tapped off' i.e. measured.

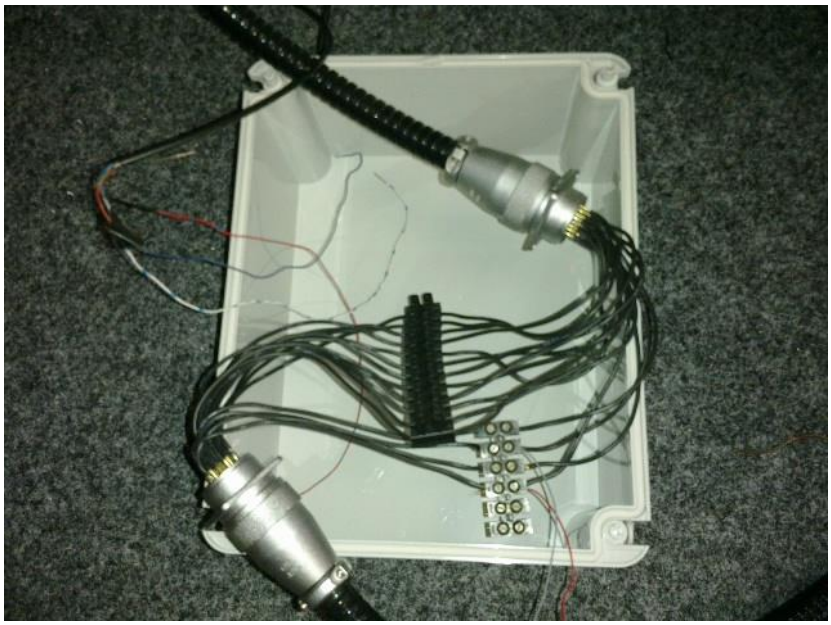


Figure 4.7 Alic-1 Servo Voltage Tapped Off at Connecter Terminals

It was not known whether the EDM machine as a whole would continue to run if its servo motor were to be disconnected, hence this arrangement made provision for the motor to remain connected during testing, if desired. I.e., the EDM servo mechanism would be producing a sort of mirror image of what the CNC motor was doing in terms of up/down movement, but it would not be physically driving the electrode, which instead would be directly fixed to the Z axis travel of the CNC machine.

Once this controlling voltage (-10V to 10V) was obtained from the EDM machine, the next step was to produce a proportional voltage to drive the CNC servo motor. Initially an analogue signal conditioning approach was used, in conjunction with an off-the-shelf motor driver, which provided a pulse-width-modulated (PWM) bipolar actuating voltage to the servo motor. The arrangement implemented is shown in Figure 4.8 below. The op-amp signal conditioning circuit (to suit the motor driver) can be seen in Figure 4.10 overleaf. In the diagram, either the EDM machine's electrode actuator (Servo Head) is used, or the CNC motor is used, together with the external motor driver and signal conditioning circuit.

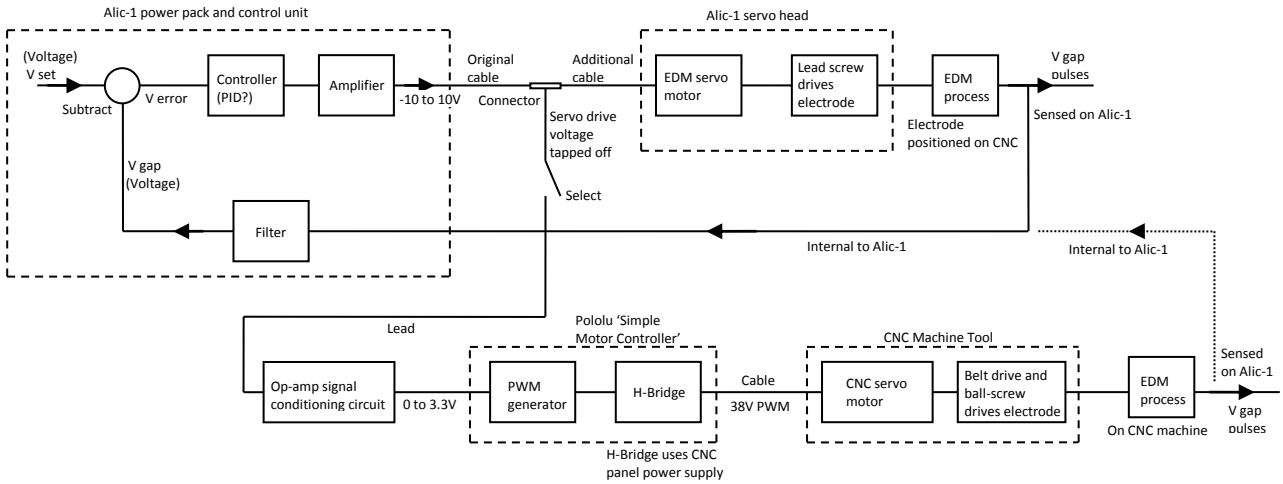


Figure 4.8 Arrangement for use of Signal from EDM Machine with External Motor Driver and CNC Servo Motor

The motor driver purchased was a Pololu 'Simple High-Power Motor Controller', model 24V12, shown in Figure 4.9 below.

Figure 4.9 Pololu Simple High-Power Motor Controller (24V12) [32]

Designed essentially for hobby-robot use, one of the controlling input signal types was a 0 to 3.3V analogue signal. This would usually be provided e.g. by a potentiometer connected to a 'joy-stick' or similar manual control device. In this case, however, no potentiometer would be used and instead the controlling voltage would come from the op-amp circuit, the sole purpose of which is to convert the -10 to 10V signal from the EDM machine's servo motor drive to a 0 to 3.3V input signal for the Pololu motor drive. Note that the Pololu input signal is not symmetrical about zero volts, i.e. the mid-point voltage (1.65V) represents zero action, 1.65 up to 3.3V represents the range (0 to 100% PWM duty cycle) of 'forward' voltage control, and 1.65 down to 0V represents the range (again 0 to 100% PWM duty cycle) of 'reverse' voltage control. Thus the op-amp circuit had to alter the range of the signal (i.e. provide a suitable gain) as well as shift the signal level i.e. provide an offset. In addition, since the Pololu drive's 0 to 3.3V input was not 5V tolerant, the circuit had to provide a clamping function to prevent a voltage higher than 3.3V from reaching the Pololu drive.

The circuit of Figure 4.10 below was used to achieve the signal conditioning required. The lower part of the circuit is however for polarity detection, which was required for a different method of control, namely when using the G-Code commands in Mach-3 (discussed later). The function(s) of each op-amp is indicated on the circuit diagram.

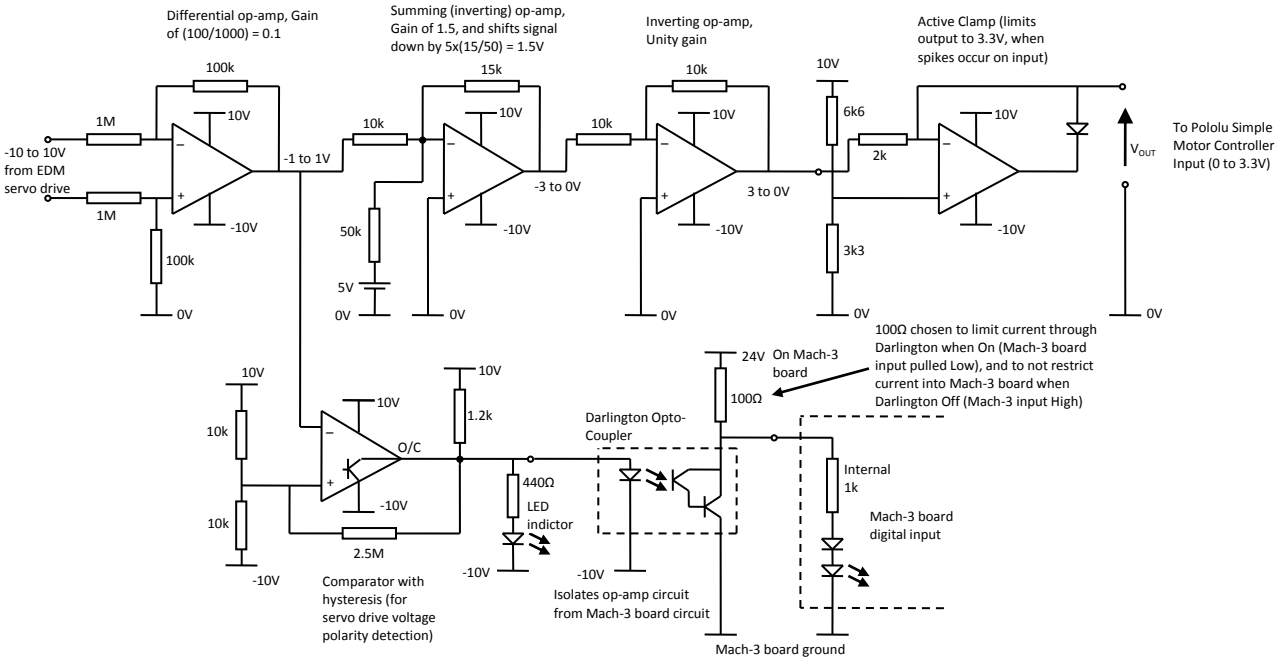


Figure 4.10 Analogue Signal Conditioning Circuit for Pololu Servo Motor Driver

Figure 4.11 below shows the 0 to 3.3V output signal (green) from the analogue circuit, superimposed on the -10 to 10V input signal (turquoise) coming from (i.e. tapped off) the EDM machine's servo drive. One can see that the voltage range has been reduced and offset, and that voltage spikes on the input signal have been clipped by the clamping circuit. The particular graph shown was captured while the EDM machine's own servo motor was in operation.

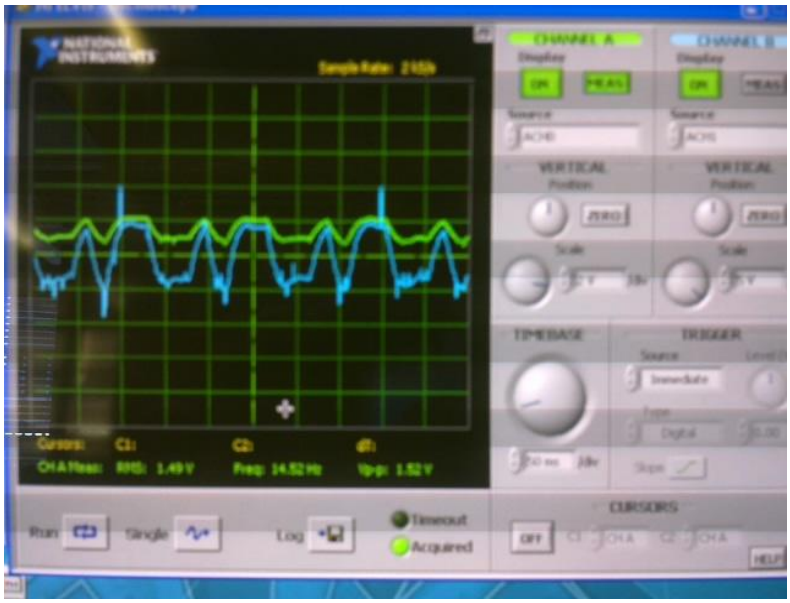


Figure 4.11 Input (Turquoise) and Output (Green) Wave Forms of Analogue Signal Processing Circuit

The Simple-Motor-Controller is to some degree programmable via a USB cable. For example, maximum output voltages (i.e. maximum PWM duty cycle) can be set (i.e. limited), and a dead-band can be created around the midpoint voltage, in which neither forward nor reverse action occurs. One concern with using the Pololu device is that its minimum update time is one millisecond. This would create a slight delay, or phase lag, between commanded changes in PWM duty cycle % and/or polarity (motor direction), and the execution of such changes. This could lead to unstable control due to the phase lag induced (depending on how significant 1 millisecond is relative to the natural period of oscillation of the control response).

Figure 4.12 overleaf shows a typical user interface set-up page, of which there are a number.

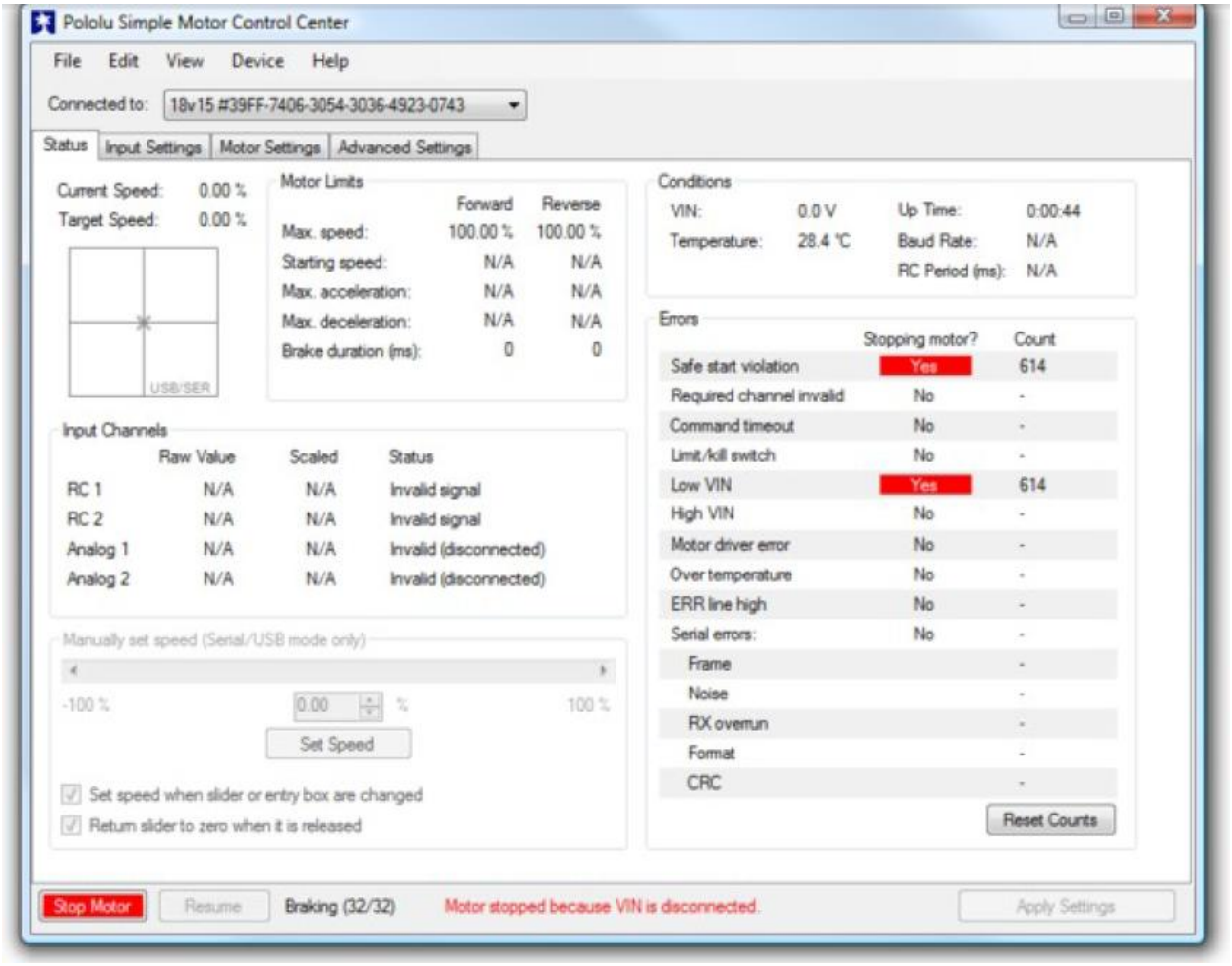
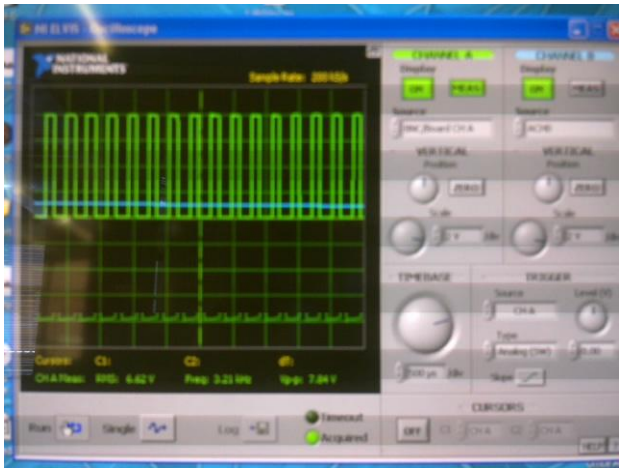
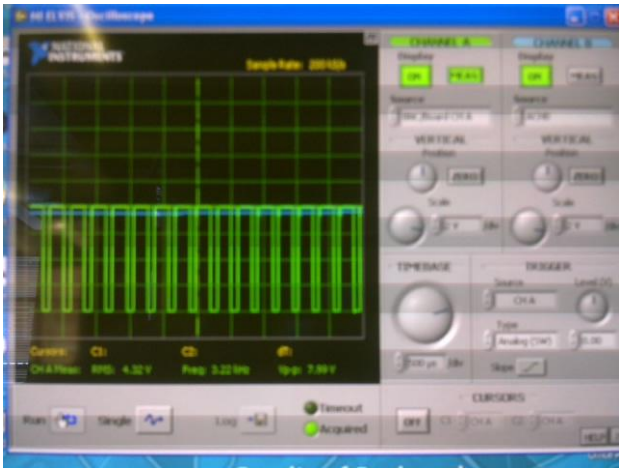


Figure 4.12 Typical Pololu Simple Motor Controller User Set-Up Page [32]

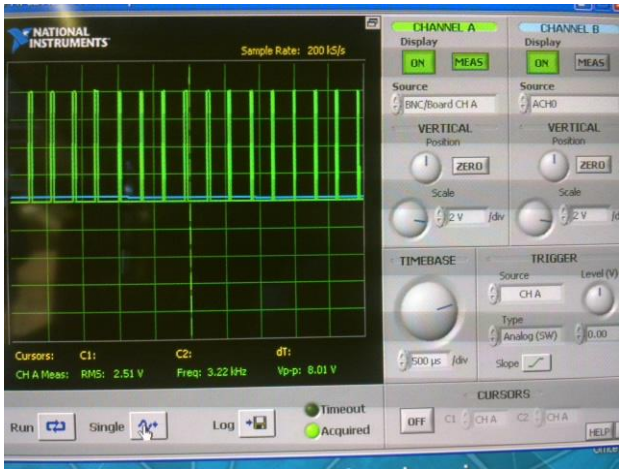
The power supply to the Pololu drive was obtained from the 38V supply in the CNC panel, which was already the supply for its servo motors (via the Gecko drives, which were now by-passed). The Pololu controller output (PWM), together with the -10V to 10V input signal from the EDM servo drive, is shown in Figure 4.13 overleaf. (a) and (b) (and likewise (c) and (d)) depict opposite motor directions, as (a) is positive and (b) is negative (since the Pololu output is bipolar, i.e. the PWM signal is after the H-Bridge). The turquoise line is the analogue signal (with 1 Division = 10V). In (c) the PWM duty cycle can be seen to be near zero, with the analogue value correspondingly small; in (d) the duty cycle is much higher, well over 50% (100% would correspond to one division (10V), for the turquoise analogue signal).



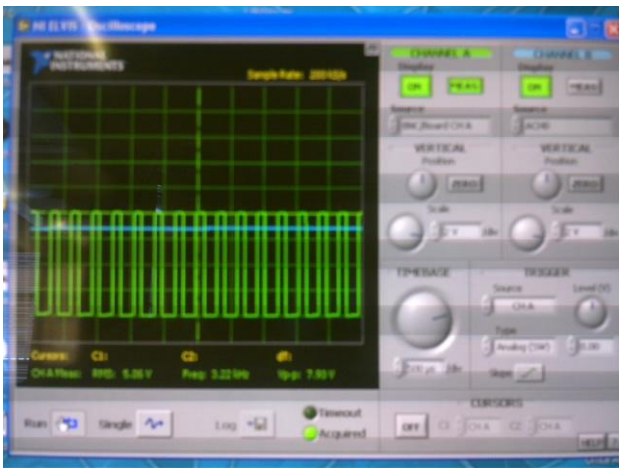
(a)



(b)



(c)



(d)

Figure 4.13 Pololu Simple Motor Controller Output

A variety of settings on the Pololu interface were tried, including altering maximum allowable speeds (which could be different for Up vs. Down movement), adjusting sensitivity (likewise could be different according to direction), and adjusting the amount of ‘braking’ during the Off time of the PWM signal (vs. ‘free-wheeling’).

The results of the testing with the analogue circuit and Pololu drive was that, although EDM did occur, a large amount of hunting occurred, such that the pointer on the Voltmeter on the EDM machine front panel displaying the gap voltage could be seen to be oscillating between nearly zero (meaning mostly short circuits are occurring) and nearly 100V (meaning mostly open circuits are occurring, as 100V is the voltage the meter displays when no sparking is occurring e.g. when the electrode is far from the workpiece during jumps). Under normal sparking conditions (i.e. with the EDM’s own

servo motor), the voltmeter is seen to be quite stable at about 50V (depending on the gap voltage setpoint – the voltage was seen to range between 40V and 60V, depending on the setpoint which could be adjusted on the EDM front panel). The frequency of oscillation of the hunting (over- and undershooting) was about 5 Hz (significantly slower than when using the EDM machine's own servo motor, which was about 20 Hz).

The reasons for the imperfect control are many, the most significant likely being the moving mass of the axis and the compliance due to the belt drive, and possibly also the one millisecond delay (update speed of the Pololu drive). Also, the amount of control signal manipulation occurring in the EDM machine itself, and the amount of fine tuning for the characteristics of its particular motor and lead-screw arrangement (e.g. torque vs. speed), are uncertain (its schematic diagrams were not very clear and were difficult to follow; there were however many op-amps present which are probably for gap voltage filtering and for PID control, and a gain ('servo speed' in the Alic-1 documentation) adjustment is present on the unit's front panel). Probably the EDM machine's motor arrangement produces less electrode movement for a given torque (which is beneficial in this instance, in terms of the need for high acceleration and fine motion). Certainly it was clear that the CNC axis' movement is significantly larger than the EDM machine's ram movement, for a given value of the -10V to 10V servo voltage signal. Thus it is likely that the control is optimized for the EDM machine's own servo motor arrangement, and hence would not be ideal for a different arrangement. So although the gain adjustment knob is present on the unit's front panel such that it could accommodate a variety of electrode masses, it may be related internally to other parameters e.g. the gain of the derivative component in PID control, and the ratio of these is as important as the amount of each. Also the documentation makes mention of a test point where 'back-emf', it claims, can be measured. This would suggest that a form of control whereby the speed of the motor is instantaneously compared to a commanded speed, to obtain an error signal for the control loop, is present (by virtue of V=IR, if the instantaneous applied voltage to the armature is known, and the armature's resistance is known (and its inductance is negligible), and the armature current is measured, the motor speed can be deduced if the back-emf constant k_B is known). This kind of control requires fine tuning to the motor being used, thus again good control cannot be expected when using a completely different motor. It is also uncertain whether lead/lag compensation is present in the EDM machine's control system.

4.5 Conclusion to Chapter 4

The control stability of the original EDM machine has been investigated, for use as a reference. The use of an LVDT position sensor for this purpose has been described, as has the implementation of methods of integration that use the EDM machine's controller directly. An analogue signal conditioning circuit has been developed as necessary for some of the methods, and an off-the-shelf motor driver having PWM output has been incorporated into the system as necessary for some of the methods.

The outcome of the test of the CNC servo motor being controlled by a voltage signal taken from the supply to the EDM machine's motor was that control was less than ideal, with too much oscillation present. In Chapter 5 following, the investigation concentrates on whether, by using G-Code commands and the Mach-3 software, satisfactory control could be achieved, again using the Z axis of the CNC machine.

5. Integration using Mach-3 and EDM Machine Servo Drive Voltage Signal

Since the outcome of the method of integration described in the previous chapter was that control was less than ideal, with too much oscillation present, the next investigation then concentrated on whether, by using G-Code commands and the Mach-3 software, satisfactory control could be achieved, again using the Z axis of the CNC machine. The thinking was that although the direct control method described above was not successful, the use of the position commands when using the G-Code could in effect slow the process down and allow for more precise, controlled, movements (less overshooting). The PID control parameters of the Gecko drives of the CNC servo motors are already tuned for the particular motors they serve, thus if a position can be commanded, and can be achieved fairly quickly, stable control might be possible.

Two options were considered for this investigation. For both, it was easier to use the Incremental position command mode as opposed to the Absolute mode. Likewise, both options rely on a knowledge of the polarity of the voltage which would have gone to the EDM servo motor, i.e. they rely on a knowledge of whether the motor is being told to advance the electrode, or to retract (i.e. whether the gap distance is too large, or too small, essentially).

5.1 Using Mach-3 and EDM Machine Servo Polarity

One option was that, while a certain polarity is being detected from the EDM control system (say commanding ‘advance’ of the electrode), then specified small Z axis steps would be continuously taken in the appropriate direction (downwards if ‘advance’ was commanded), and the reverse would occur when/while the other polarity was detected. I.e. if the control EDM servo drive voltage was say positive (0 to 10V), steps would be taken in a certain direction, and while it was negative (0 to -10V), steps would be taken in the opposite direction. For this option the frequency of the steps would remain fixed, i.e. the only determining factor is the polarity of the EDM servo drive output. The other option was the same except that not only the polarity of the voltage would be considered, but also the magnitude of the voltage – the commanded step frequency then being dictated by the voltage magnitude. This amounts to adjusting the feed rate in the G-Code program, according to the voltage size.

5.1.1 EDM Machine Servo Voltage Polarity Detection

To effect the first strategy, it was first necessary to determine the polarity of the EDM servo drive output; this then needed to be somehow communicated to the G-Code program running in Mach-3. Finally this information needed to produce the appropriate change in the step direction commanded in the G-Code program. Figure 5.1 below shows the circuit used to determine the polarity and communicate this to one of the input pins on the Mach-3 breakout board. The upper part of the circuit was for signal conditioning for the previous control method discussed, and (except for the first op-amp) is not part of the polarity detection and Mach-3 input driving circuit.

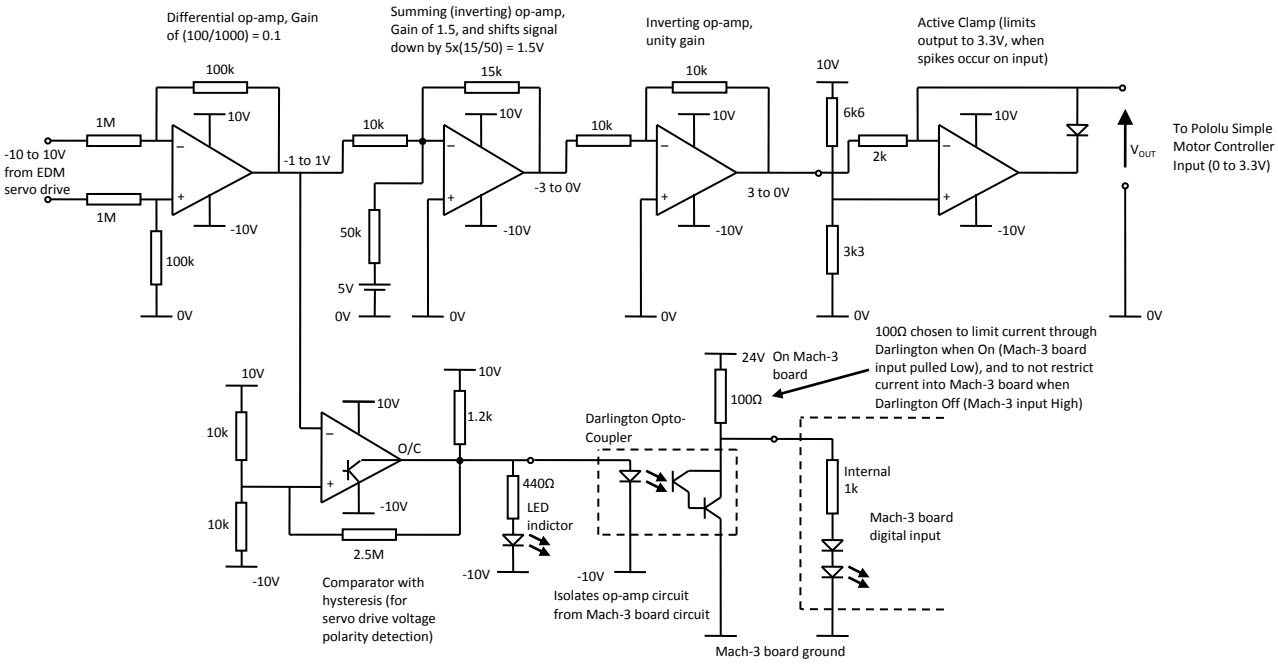


Figure 5.1 Circuit for EDM Machine Servo Voltage Polarity Detection

A comparator was used to check whether the voltage was above or below zero. Since there is a lot of noise on the voltage signal, it was important to implement a certain amount of hysteresis on the comparator, such that multiple triggering at the zero crossing of the main signal did not occur. It was thought that this would likely create more stable control, than if many fluctuations were occurring. The circuit was designed such that a positive going signal would trigger a high state from the comparator when about 0.1V is reached; a low state would only be triggered when the negative going signal reached about -0.1V (these amounts represent 10% of the signal range, which is -1 to 1 at this point). Thus there was a 0.2V hysteresis band, a sort of a dead-band in

this instance, in terms of change of output. One would not want the hysteresis band too small, as multiple triggering would occur, unnecessarily, and one would not want it too large, otherwise it would create a lag in the response of the system. A 0.2V band was thought to be a reasonable compromise, bearing in mind that the total range of voltage is -1V to +1V at this point (although the signal does not necessarily always have to range all the way to the extremities).

The state of the comparator, which indicates the polarity of the servo drive voltage, then needs to be communicated to the Mach-3 breakout board (which is connected to the parallel port of the PC running the Mach-3 program). The comparator output ('open collector' type) was used with a 'pull-up' resistor to drive an opto-coupler transistor's LED emitter, the transistor being used with another pull-up resistor, to provide or not provide current into one of the digital input pins on the breakout board, i.e. driving it high or low. The breakout board's inputs are optically isolated, with a 1k resistor in series with the LED emitter of its own isolating opto-coupler. Although the breakout board has this isolating device, the LED's ground is common with the rest of the breakout board's ground, hence the use of the additional opto-coupler to simplify isolation as now the break-out board ground and the ground of the comparator circuit were completely independent. This was important as the power supply to the comparator circuit was from the ELVIS DAQ development board (and the ELVIS board is connected to the PC running LabVIEW), whereas the Mach-3 breakout board's ground was from the CNC machine's panel; if these were tied together, a ground current loop could occur which may have caused problems. A separate LED was connected between the comparator's output and ground, for visual confirmation that polarity changes were being detected.

5.1.2 Mach-3 G-Code and Macro Program

In Mach-3, the state of individual input pins on the breakout board can be seen/queried, and this determined whether the G-Code program would advance or retract the electrode.

The simple G-Code and Macro program developed is shown below. The programs are typical, but a variety of variations were tried.

```
G-Code (Macro program call):          ! Runs in Mach-3 main program execution window

G91                                   ! Use Incremental mode

G0 X 5                                ! Rapid Feed X axis 5mm forward (to check whether
                                        G-code program is working)

G0 X -5                               ! Rapid Feed X axis 5mm back

M90 001                               ! Call Macro 'M90 001.m1s' from 'Macros' files
                                        folder

Macro M90 001.m1s :                   ! Resides in Macros folder

G0 Y 5                                ! Rapid Feed Y axis 5mm to left (to check whether
                                        Macro is working)
G0 Y -5                               ! Rapid Feed Y axis 5mm back

Do                                    ! Start 'Do… Until loop
    If IsActive(INPUT1)  Then         ! Detect whether servo polarity positive, using
                                        input port 1

        Code ("G0Z 0.005")            ! Rapid Feed Z axis downwards (i.e. advance) 5
                                        micron if polarity positive

    Else
        Code ("G0Z -0.005")           ! Otherwise Rapid Feed Z axis upwards (i.e. retract)
                                        5 micron

    End If

    While (IsMoving())                ! Don't continue until Mach-3 is not busy (i.e. no
                                        pulses being sent to Gecko drive)

    Sleep(10)                         ! Wait 10 ms

    Wend                              ! Ends the check whether busy

Until IsActive(1003)                  ! Stop 'Do… Until' loop when 'Program Stop' button
                                        is pushed (i.e. clicked with mouse)
```

The programs operate as follows. The main program (G-Code program) is loaded in Mach-3. First <G91> is specified, which puts the machine into the Incremental motion mode, i.e. travel commands are specified as a distance of movement from the current position (whereas in the Absolute mode (<G90>), travel is specified by the co-ordinates to be moved to, relative to a fixed datum or 'Home' position). Then a simple move command for the X (horizontal) axis was specified, just to check visually that the program

is executing. A call is then made to a ‘Macro’ program (M90 001.m1s in this case) which was written for the application. The Macro must be stored in a sub-folder under the ‘Macros’ folder under the Mach-3 program folder, for addressing by the G-Code program, and must have the form ‘M*.m1s’. Macro’s in Mach-3 utilize Visual Basic Scripting language commands; also some standard functions, based on VB Script commands, have been written for Mach-3 and are available for use in such programs.

The <Do> command starts a <Do… Until…> loop. Detection of the state of the relevant input pin on the breakout board has the form of <IsActive (INPUT1)>, where pin 1 is the relevant input pin being driven High or Low according to the polarity of the EDM servo drive. Thereafter a basic <If… Then… Else… End If> statement is used, such that if pin 1 is High then a small upwards step would be taken, and if not (i.e. if Low) then a small downwards (since negative distance specified) step would be taken. The meaning of High and Low can be defined in the Mach-3 set-up screen, where the particular pin can be defined as ‘Active High’ or ‘Active Low’. The <Code(“…”)> function transfers the text in the inverted commas to the main G-Code program calling the Macro, which then immediately executes the text command, in this case <G0 Z 0.005>). <GO> indicates a ‘Rapid’ linear travel command (where ‘Rapid’ feed speed is defined elsewhere in the Mach-3 set-up, and is essentially set as fast as the particular axis can move without shudder), and the ‘Z’ indicates the Z (vertical) axis. ‘0.005’ indicates the distance to be travelled, in mm (thus 5 microns in this instance). The <While (IsMoving())… Wend> command detects whether any G-Code movement commands are currently being executed by the main program, i.e. whether pulses are currently being sent to the Gecko drives serving the servo motors. This command is important to use here, as if not, travel commands would be continuously sent to the main program (since the Macro program commands are in a <Do… Until> loop), and would buffer so fast that they cannot be executed quickly enough, i.e. they would buffer faster than the pulses are being sent to the Gecko drives, and Mach-3 would run out of memory, and the program would crash. Also, even before crashing, the program would be a useless in that by the time any particular travel command is executed, the polarity of the voltage of the EDM servo drive would almost certainly have changed, thus creating movement in the wrong direction in terms of keeping the electrode the correct distance above the surface of the workpiece.

It was not certain whether the <While (IsMoving())… Wend)> command allows a short time interval before the sending of the last travel pulse and the continuation of the

program. If not, it is possible that pulses can be sent, say decrementing the Gecko position command counter, when previous pulses say incrementing the counter have not yet been obeyed in terms of physical movement of the servo motor. This is not necessarily a bad thing however, in that if such decrement pulses are being sent, it means that the control program wants to change the direction of the servo motor at that point, whether or not previous increment commands have been physically obeyed (the inertia of the motor naturally prevents it from moving instantaneously). However, it was uncertain as to whether this may cause problems in terms of the action of the Gecko drive. Conversely, if a delay is inherently created after the last pulse is sent when using the <While (IsMoving())… Wend)> command, then this delay may be unnecessary and may cause an undesirable phase lag in the control of the system. Thus, the <Sleep (*)> function was experimented with, by itself or in conjunction with the <While (IsMoving())… Wend) command. This command simply forces a time delay in the execution of the program, where the time is specified in milliseconds. It was found that a (relatively) short time delay (on the order of 200 ms), by itself (i.e. not in conjunction with the <While (IsMoving())… Wend)> command, produced better results than using the <While (IsMoving())… Wend)> command by itself, or in conjunction with a <Sleep (*)> command.

The <Until IsActive(1003)> statement near the end of the Macro completes the <Do…. Until…> loop, finishing the Macro when the Stop button on the Mach-3 front screen (HMI) is pressed (clicked; ‘1003’ refers to the Stop button). Usually the program would return to the main G-Code program that called the Macro, but in this instance, since the loop finished only when the Stop button was pressed, the program would stop completely as the Stop button automatically stops all program execution anyway.

5.1.3 Discussion of Performance

Although a measure of control was achieved using this technique (i.e. using the G-Code and Macro in conjunction with the EDM machine’s servo drive polarity detection, to control the electrode), the EDM process as a whole was not entirely satisfactory using this approach. It was found that although sparking did occur, it could not be maintained continuously, with frequent short circuits (i.e. electrode too close to workpiece) stopping the continuity of EDM (it appears that when (enough) short circuits are detected by the EDM machine, it temporarily stops the sparking pulses, and waits a short time to make

sure the electrode has enough time to back off from the workpiece and settle; a lamp on the front panel of the EDM machine indicates when short circuits are occurring, and one can see the voltage dipping and then rising to the open circuit voltage shortly thereafter). Naturally this method of control is not as fast and responsive compared to driving the Z axis motor from the EDM machine's servo voltage signal, as it is based on short individual steps, with short gaps in-between; however it is inherently a more stable kind of control in terms of not causing large overshoots of the electrode. In this sense it is more akin to using a stepper motor for electrode movement purposes (which is done in some machines, but seems to be the exception rather than the rule, presumably because it is perhaps not as fast and fluent as a servo motor in terms of adjusting to the fluctuating gap conditions; also stepper motors have a lower torque-to-size ratio than servo motors, thus larger motors are required if stepper motors are to be used).

One might have thought that short, sharp steps would be evident from the Z axis motor, when using this type of control; however this was not the case. Although step-like movements could be seen to be taken, they did not appear as short and sharp as might have been expected. The presumed reason for this is that the error count (difference between commanded position and actual position, as held in the respective counters in the Gecko drive), has to be fairly high to induce a large response from the drive in terms of the voltage applied to the motor's terminals (i.e. the PWM duty cycle). In other words, the proportional (P) gain (K_p) of the Gecko drive control loop is not very high; instead, it appears that it relies quite heavily on the integration (I) component of the control action, to force changes in voltage (hence position corrections) when the position error is small (naturally the P gain is limited as if too high, excessive motor vibration (overshooting) occurs). The integration component of the control action naturally relies on the passing of some time in order for control action to 'build up'. Thus although the correct position is eventually reached, there can be a short (but significant in EDM) time delay involved.

The apparently large change in PWM duty cycle required for a step to be taken would mostly be due to friction (as opposed to inertia), since the ball-screw was self-locking. I.e. it required a 'downwards' voltage to be applied to achieve downward movement, as opposed to just a reduction in 'upwards' voltage, which would be the case if it was not self-locking, i.e. if a tendency for downwards motion due to gravity always existed and would need to be continuously counteracted by a voltage tending to lift the electrode. So for a step to be taken requires a voltage change large enough to overcome friction in

each direction. Once moving, an axis experiences less friction since the co-efficient of static friction is higher than dynamic, and thus is likely, due to inertia, to overshoot (and the PWM duty cycle which has just peaked would have to be immediately significantly reduced to oppose the tendency to overshoot – even possibly requiring a polarity reversal just to stop the motion before a step command is given in the other direction which would naturally change the polarity anyway).

If larger step sizes were specified in the Macro program, response was definitely seen to be more immediate (the position error is immediately larger when the larger step command is given (before the motor has time to respond); the proportional component of the control action is immediate as it depends only on the size of the error, not integration over time). However the specified step size had to be significantly larger than amount of position variation that the EDM process ideally allows (say 10 to 15 microns either side of a midpoint, for smooth control with continuous sparking) – step commands had to be on the order of 20 or even 50 microns for fast, sudden steps to be taken. This naturally caused overshooting relative to typical position variation in EDM, and the pointer of the Voltmeter on the machine front panel could be seen to vary widely; even then, short circuits did occur and cause the process to stop momentarily because of the large overshoot). Again, if the motor did not have time to fully obey the relatively large step commands before a command came to reverse direction, this would in fact be better, as less overshooting would occur.

To get an indication of the fineness of EDM control, it should be noted that the minimum step distance (i.e. resolution) for the CNC Z axis motor is about 4 microns (not exact, due to a strange gear ratio used on the Z axis). This resolution is based on the number of divisions of the encoder discs (500 cycles per revolution, thus 500 X 4 = 2000 individual pulses (encoder state changes) per revolution, as one cycle comprises 4 encoder pulses in a quadrature encoder, and based on the lead distance of the ball-screw and ‘number of starts’ i.e. parallel threads and the screw, and based on the gear ratio between motor output gear and ball-screw gear. This puts the potential range of error before a step is taken, at about 8 microns, as once a step is taken one direction, the encoder will not change state until an additional 4 microns of movement has occurred in that direction, or until 4 microns movement has occurred in the opposite direction (i.e. a quadrature encoder has a sort of hysteresis built in, in terms of detecting state changes when direction is reversed). Thus one is hoping for very few individual

steps to be taken either side of a midpoint, during an EDM process. This relatively large resolution of the CNC machine's axis movement meant that it was always going to be difficult to achieve stable control using its motor. The situation could undoubtedly be improved by changing the gear ratio of the Z axis, i.e. smaller gear on the motor, and larger gear on the ball-screw, thereby creating finer movement resolution for each step pulse, but this would involve mechanical changes to the machine, which was not desired. The Gecko drive has a 'step multiplier' option whereby for each pulse received from the breakout board, up to ten encoder steps are commanded. Care was taken to ensure that the multiplier options were not selected on the Gecko drive, i.e. a pulse ratio of 1 to 1 was used, giving the minimum resolution distance.

The exact distance specified for each step in the Macro was experimented with to see whether control could be improved; when 5 microns was used, the concern was that sometime a double step may occur, since the resolution of the axis was about 4 microns. If 4 microns was used, and the actual resolution was slightly more than 4, then there would be some instances when no action would occur when the program issued a step command, as the 4 micron distance would lie within the resolution distance. Naturally the process would nevertheless accommodate these eventualities, as if a double step is taken which possibly results in too much movement one way, the electrode will immediately be commanded the other way due to the monitoring of the servo drive voltage; likewise, if no step is taken when one is commanded, a step will be taken anyway on the next iteration of the <Do… Until…> loop. But, the smoothness and consistency of the control may be affected.

The G-Code program as shown above does not allow for a specific depth of EDM to be reached for the automatic stopping of the program, but this could easily be addressed by monitoring the value of the Z axis digital readout (DRO) in Mach-3 and using a specified value as a stopping criterion. Note that it is entirely unnecessary to stop the EDM machine itself from producing its pulses when a certain depth is reached, as the process stops naturally, as once the electrode stops moving, the workpiece material nearby gets eroded until the distance between it and the electrode is large enough (all around the 'contact' region between workpiece and electrode) that no sparking occurs across the gap (i.e. no dielectric breakdown occurs, even though the EDM machine continues to fire (pulse) it's MOSFET). An interlock could be considered though to turn off the EDM 'spark generator' and the dielectric fluid pump, when the G-Code program reaches a

stopping point. This would probably best be done in hardware (i.e. by changing the Alic-1's Start and Stop pushbuttons to include a relay function, which could be activated from the Mach-3 breakout board (an output pin(s) can be driven High or Low according to the G-Code program, which could activate a relevant relay(s)). However, output pins are in short supply on the Mach-3 breakout board since it uses the parallel port of the PC. Another breakout board which is uses USB communication is however available, which would make such interlocks more easily realizable.

5.2 Using Mach-3 and EDM Machine Servo Polarity and Magnitude

The second option using G-Code considered, as mentioned above, would involve the use of the size of the EDM servo voltage in addition to its polarity. However, in order to effect this option, ideally an analogue input would be required on the Mach-3 breakout board (which the board being used unfortunately did not have – it has analogue and digital outputs, but only digital inputs). A way to overcome this would be to use the digital input designed to detect the speed of the spindle (there is a function in Mach-3 whereby the output to the spindle can be adjusted via a PID loop, to enable to the spindle speed to exactly match a specified speed, by feedback measurement of the actual spindle speed). This input detects the frequency of pulses received from (usually) a reed switch or Hall-Effect sensor, switched as a magnet or magnets on the spindle shaft pass the sensor. Thus if a (digital) pulse frequency proportional to the (analogue) servo drive voltage could be produced and read into this digital input, effectively the size of the voltage signal would be available as a variable in Mach-3. Voltage-to-Frequency converter IC's are readily available and could be used for this purpose. The servo voltage, now represented as a frequency (a number in Mach-3), could then be used in a similar Macro to the one shown previously, but instead of the "G0" command (specifying rapid travel), a "G1" command could be used, with a feedrate specified (by the frequency variable or a multiple thereof) immediately before it (which would be updated with each iteration of the <Do… Until…> loop). "G1" specifies a normal travel command, and Mach-3 uses whatever feedrate is currently specified (i.e. last specified) to execute it. The feedrate would be specified using the "F" command, followed by the <Get...> function, where the relevant variable storing the frequency would appear adjacent (after) the <Get> function.

An alternative to using voltage to frequency conversion for the digital representation of the analogue voltage, would be to use a binary word (say 3 bits since inputs are limited on the breakout board) by using a suitable analogue to digital converter (IC), and reading the word into the Mach-3 breakout board using the appropriate number of inputs. Although the quantization error would be large ($1/2^3$ = 12.5%), the 3 bit word would still be a rough indicator of the analogue voltage, and again, the control system would to some degree by nature be self-accommodating.

However, it was felt that this approach would not likely be much, or any, better than the method previously described (even though it makes more sense from a 'Proportional' control point of view), since it had already been found that the 'step' response from the RMT motor was not very fast (for small steps), even when a rapid travel was specified. Thus this option was not investigated further.

5.3 Multi-Axis Implementation using Mach-3

The above two options could be applied also to multi-axis EDM, simply by indicating two (simultaneous) axis travel commands in the Macro (Z and X, or Z and Y), again using the Incremental G-Code programming mode (and indicating the negative of these travel commands for when the servo voltage changes polarity, i.e. for the <Else…> part of the Macro). This would produce a linear trajectory of EDM, at an angle to the vertical.

Programming a suitable Macro in Absolute positioning mode would be a little more difficult, in that one cannot just specify a specific position to go to when the servo drive voltage is positive, and another to go to when it is negative, as the distance between these two points would need to be very, very small or fouling will occur.

The ideal solution for the control process would be to 'get inside' the Mach-3 interpolator. I.e. if a (time) variable exists which steps the G-Code program through the linearly (or circularly) interpolated points between the end points specified of a given travel command, then that variable could be just incremented or decremented according to whether the ED machine is dictating advance or retract (in terms of its continuous detection of the gap condition). Access to the interpolator did not appear readily achievable, and at this point attention turned instead to methods of producing co-ordinated movement pulses by an external source, i.e. not Mach-3 itself (although if a PC

based solution, the program could run on the same PC as is running the Mach-3 program). Also, it was felt that if a different servo drive was used (a more programmable one than the Gecko drive), then parameters could be better optimized for the task at hand, and the type of control could also be improved.

5.4 Conclusion to Chapter 5

This chapter described how EDM could be achieved on the RMT by making use of the CNC controller and hardware, combined with a gap condition signal from the EDM machine's controller. The Mach-3 CNC programming language was used, and in terms of hardware, the RMT's PC, breakout (interface) board, Gecko servo drives, and servo motors themselves, were used.

This method of integration proved for the most part unsatisfactory, thus attention is turned to the final category of integration described in Chapter 6 following, namely integration making use of an external controller,

6. Integration Using External Controller

Using G-Code commands proved unsatisfactory as discussed in the previous chapter. Without access to Mach-3's interpolator, the next best form of control would likely be by means of sending step and direction pulses to the Gecko drives directly, from some other source. In that case, no waiting would be required to ensure that G-Codes are not buffering too fast, nor would any 'While IsMoving' type function be required. Specific pulses could be sent to the Gecko drives at will, resulting in a more precise and responsive system.

One perceived shortcoming of the Gecko servo drive used for the CNC machine is that it apparently responds only to position commands, i.e. requires a position (counter) error to effect a change in PWM of the voltage applied to the armature, whereas it may be preferable to use a drive which takes commanded velocity into account in the control action, not just the integration of velocity, which is position. This would create a more immediate response, since as soon as the frequency of pulses being received by the drive is detected (which is almost instantaneous – requires two pulses minimum) control action could be taken (pulse frequency of course being proportional to velocity, i.e. rate of change of commanded position). This as opposed to 'waiting' for time to pass as the drive counter is incremented, to cause a significant change in control action. Basically, if pulse frequency response is implemented, the drive can respond with a step change in control action when a step change in commanded 'setpoint' (frequency) is specified, by virtue of a proportional gain acting on the frequency, i.e. detected rate of change of pulses. Whereas, with position only control, it is not really possible to achieve an immediate step response – pulses can only be sent to the drive at a certain maximum frequency, and this frequency has to be integrated over time (e.g. a counter has to incremented), to achieve a control output change. I.e., response is more like a ramp function when the control input (command) is a (frequency) step function.

To implement this form of control, an external controller would be required instead of the Gecko drive, which is really a controller and (PWM) amplifier in one device.

6.1 Linear Actuator

Even with the improvements mentioned above in mind, it was felt that good EDM was not likely to be achieved given the inertia of the axes and the flexibility of the timing belts used between the motors and the ball-screws. Thus, it was decided that a small actuator should be purchased, which could be mounted on one of the axes of the CNC machine. This option had from the start been in the author's mind as a likely eventual requirement in order to achieve good EDM. Ideally two actuators are required, for the proper demonstration of multi-axis EDM (or possibly the original electrode servo head of the purchased EDM equipment could be used together with one linear actuator). Part of the idea of the use of 'independent' linear actuators is that, in theory, only the power pack (spark generator) and di-electric fluid tank & pump need be acquired from an EDM equipment manufacturer (or can be relatively easily built – 'Build your own EDM machine' books are available, e.g. 'Build a Pulse EDM', by Ben Fleming [33]). This is also in line with the modular approach (the power pack is a module, the tank & pump is a module, and the actuator with electrode is a module).

A relatively short stroke moving coil linear actuator, with built in quadrature encoder, was decided upon. From an integration of EDM onto an existing CNC machine tool point of view, it was thought that a sort of 'inch-worm' approach could be used, for instances when a long sinking distance of EDM is required. This process can be described as follows (assuming for now that the actuator is to be mounted vertically, on the Z axis of the CNC machine, its shaft extending downwards for sinking EDM). The process would start with the actuator retracted. The actuator would then slowly extend, performing the EDM, while the Z axis remains stationary. Upon reaching the end (or near the end) of the actuator's stroke, the actuator could be made to retract completely, or nearly completely; thereafter the Z access of the CNC machine could be instructed to travel (quickly) the same distance, or slightly less, than the stroke of the actuator. Then the process could begin again, with the actuator slowly extending while performing the EDM, and so the process could continue the full length of the Z axis travel. If two such actuators were purchased, the moving part of one could carry the stationary (mounting) part of the other, at 90°, so that multi-axis machining could take place. For long distances, the 'inch-worm' approach as described above could be used in a similar, co-ordinated, fashion. When one actuator reached the end of its stroke, both actuators would need to be commanded to retract; and the two CNC machine axes being used

would then each have to step forward the same amount as its corresponding actuator was extended prior to the retraction was commanded. One actuator will invariable reach the end of its stroke before the other, unless a 45° angle is being EDM'ed.

A Piezzo-electric actuator also was considered for the inch-worm EDM. An advantage of such an actuator is that motion is quick and rigid; also, if a specific voltage is applied to the crystal, a known distance should be moved through thus if voltage can be controlled, distance can be controlled. The disadvantages of such a device are cost, and very short stroke capability, even with a large Piezzo-electric stack actuator – meaning that many more frequent steps must be taken using the CNC machine's own servo motors.

The linear actuator purchased is a LCA25-010 from SMAC, having a 12mm stroke and rated up to 48V supply. A typical actuator from the LCA25 series is shown in Figure 6.1 below.

Figure 6.1 SMAC Moving Coil Linear Actuator [34]

As mentioned above, the actuator is a moving coil type, and has a built-in encoder. The device is essentially the same as a DC motor, but with linear not rotational motion. The standard encoder has a 5 micron pitch, but for the application at hand, it was deemed worthwhile acquiring the 1 micron pitch option (considering that in an ideal EDM process, deviation of the electrode from either side of an (imaginary) midpoint is only about 10 (perhaps 15 maximum) micron. Especially if encoder pulse frequency is desired to be measured, one wants ideally many pulses to be present within a 15 micron range, not just 3 or so. A 0.5 micron encoder was also available, but the addition cost was

substantial. For the linear encoder, the pitch size being mentioned is not for a cycle of 4 pluses, but is for each pulse (state change) when the device is moving. Thus there are 12 000 pulses (technically 11 999 state changes) from the encoder over the 12mm stroke, since the 1 micron pitch was chosen.

6.2 Motor Driver for Linear Actuator

The linear actuator naturally requires a driver, or amplifier, to provide (controlled) power to the armature. For low voltage applications, an analogue transistor or power op-amp solution could be appropriate, but for higher voltages a PWM (Pulse Width Modulation) type amplifier is required, to minimize VI losses that would occur in transistor based amplifiers. A driver based on the ST L298 dual full-bridge motor driver IC was available and was initially used. The driver board is depicted in Figure 6.2 below. Schematic diagrams of the ST L298 driver and its use in bidirectional DC motor control are given in Figures B1 and B2 of the Appendix B [35].

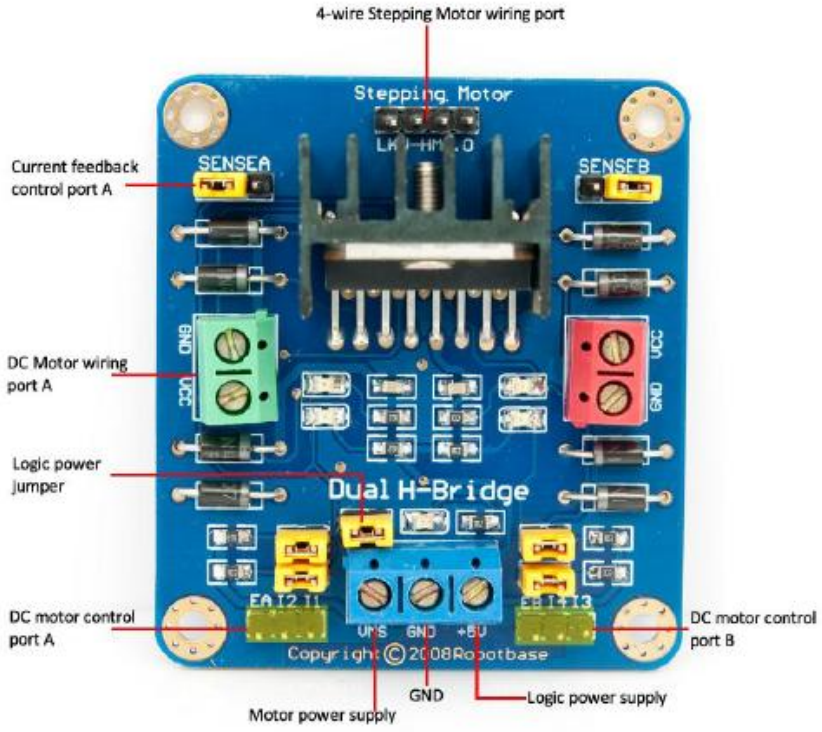


Figure 6.2 Robotbase L298 Motor Driver [36]

'Free-wheeling' or 'snubber' diodes are included (for high inductance applications, such as the coil of the linear actuator), as well as connector terminals, input protection, and indicator LED's. The driver can be used for a DC motor, in bipolar H-bridge mode, or can be used for a Stepping motor. In this instance naturally the DC motor H-bridge

connection is required. Only one port need be used, as the driver is a dual channel device, capable of supplying two independent motors. The L298 chip is mounted on a heat sink to improve current handling capacity.

The driver board has two 'Direction' inputs per channel, and one 'PWM' ('Enable' on the L298 driver) input per channel. If the one (and only one) of the Direction inputs is High, this dictates the direction of rotation of the motor, or advance/retract in the case of a linear actuator. If both Direction inputs are High, this produces a 'fast' motor stop (braking), and if both are Low, the motor is able to 'free-wheel', i.e. it produces a 'free-running' motor stop.

Towards the end of testing, when velocity control mode was used and very fast direction changes of the actuator were occurring, the current drawn became apparently too large for the bridge, and it overheated and was burnt out. An individual ST L298 chip was available and was used instead; however both channels were used together, i.e. in parallel, such that the current carrying capacity was higher. In this instance, the circuit was built on a prototyping board ('breadboard'), and individual snubber diodes were used. The circuit proved successful, with the IC barely getting warm (the heat sink from the damaged driver board was re-used on the L298 chip).

The circuit implementation on a prototyping board is shown below. The schematic diagram can be seen in Figure B3 of Appendix B [35].

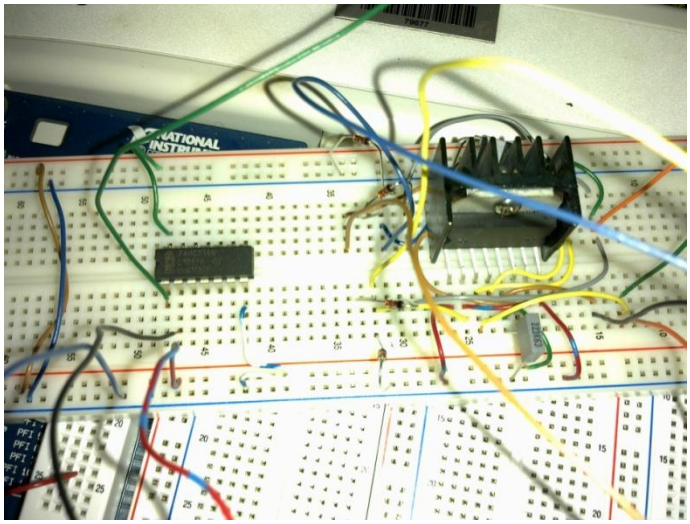


Figure 6.3 Circuit Implementation for ST L298 Full-Bridge Motor Driver, Outputs Connected in Parallel

6.3 LabVIEW and DAQ Device as External Controller

To implement the desired control improvements, i.e. the ability to dictate pulses directly, and to potentially have the drive respond to pulse frequency (representing velocity) as well as just the number of pulses accumulated, it was decided to investigate the use of the National Instruments (NI) LabVIEW software, in conjunction with a suitable data acquisition (DAQ) device. LabVIEW is a graphical programming language, based on the concept of data-flow programming, and is well suited to process control type applications, where input data must be continuously manipulated and outputs continuously produced. The NI ELVIS development board (Figure 6.4) was available at NMMU, for use with LabVIEW.

Figure 6.4 NI ELVIS DAQ Development Board [37]

The development board is essentially a DAQ device, combined with a convenient development board containing power supplies, proto-typing board ('breadboard'), various input and output connectors, and certain 'built-in' functions including a digital multimeter, function generator, and an oscilloscope (all of which run in the LabVIEW programming environment).

The external controller options require direct access to the process variable, the gap condition (distance or voltage). The use of LabVIEW and the DAQ device would also open possibilities in terms of accessing and filtering this signal.

A number of LabVIEW control programs were developed and their performance investigated, some being relatively simple, and others more complex (generally programs started as fairly simple, and became more complex as additional features and

functionality were added). To some degree the programs are modular in approach. A few of the programs developed will be described and discussed below. The programs were generally developed with the use of the linear actuator in mind (or two linear actuators), with final motor drive being simply an amplifier of sorts (i.e. not the Gecko drive); but the programs could be modified for use with the CNC machine's motors (by themselves or in conjunction with the linear actuators, or even using a hybrid approach with one axis utilizing a linear actuator and the other (perpendicular) axis utilizing the CNC machine's own servo motor). Specifically they were developed with the use of a PWM controlled H-Bridge motor driver in mind, but could in some instances be modified for use with the existing Gecko drives, which of course have their own internal PID positioning control function, i.e. they produce their own PWM signal and do not receive it as an input like a basic H-Bridge driver would.

6.4 LabVIEW Programming

LabVIEW, like any other programming language, has many standard, basic functions, which can be combined together to form more complex programs. Libraries of commonly used program modules are also available, and can be incorporated into one's program. LabVIEW programs are known as virtual instruments (VI's), and contain two screens, namely the 'Front Panel' which acts as a HMI (human-machine-interface) for the user, and a 'Block Diagram' screen where the programming takes place. Basic functions are invariably represented as graphical function blocks in LabVIEW, i.e. as icons of a sort. Blocks are connected with 'wires' which communicate various types of data from one element to another. Commonly used program routines (essentially sub-VI's) have their own front panel and/or block diagram, and can be represented by an individual icon in the main program block diagram, in much the same way as the more basic functions. Wires can likewise link these sub-VI blocks with basic functions blocks. Items on the front panel are generally either 'Controls' or 'Indicators'. An Indicator displays the value of some variable, in one form or another, whereas a Control sets the value of a variable (e.g. user input). Each element on the front panel has corresponding block on the block diagram, such that information can be transferred between the environments.

The programs developed use mostly basic functions combined by the author to achieve desired functionality, in combination with some appropriate standard sub-VI functions.

The basis of many LabVIEW programs is a ‘Do’ loop (Do While, Do Until, etc.), which is represented by a rectangular box (border) on the block diagram screen, which constantly iterates. Continuous data acquisition from external real world sources takes place inside a loop, where the relevant input port on the DAQ device is addressed inside the loop (a block representing the port is positioned inside the loop, on the block diagram). Likewise continuous output of data from the program to the external world generally takes place inside a loop, where the relevant output port of the DAQ device is addressed. For each input and output, various parameters must be set by the developer. A variety of types of analogue and digital data can be read into and written from the program. To transfer values of variables from within one loop to other parts of the program during execution, typically a Local Variable is used; this is because a wire from an element in a loop, connected across the loop’s representative border, takes on the element’s value only when the loop finishes executing, and not while it is running. A Local Variable refers to an element already in existence on the block diagram or front panel.

A Shift Register in LabVIEW is a loop in which the output of a specific variable after a certain iteration of execution automatically becomes the variable’s (starting) value for the next iteration of the loop. This is useful for generating counters for example. A so called ‘Feedback Node’ performs a similar function, but the representation is different and no loop is visible.

The programs developed generally performed three tasks, namely to create a control loop (or loops) for the EDM control process, to perform signal conditioning (manipulation and filtering), and to create the desired type of output for the chosen physical hardware device (amplifier, essentially), which would power the chosen linear actuator (or potentially any actuating motor). Generally the programs were written with an H-Bridge (i.e. bipolar) motor driver requiring a PWM input signal in mind. A number of control strategies and the programs developed to implement them are described below, along with details of electronic circuit diagrams developed as necessary to suit.

6.5 Gap Signal Acquisition and Filtering

Some of the LabVIEW programs developed could technically be used in conjunction with the EDM machine’s own servo drive voltage signal (-10 to 10V), as the control input

(actuating signal). I.e. the servo drive signal could have been used as an indicator of gap condition, as it is based on that; however, it most likely has a significant amount of signal conditioning upstream – it was probably a PID output based on the gap, even possibly containing motor back-emf in the strategy. Thus it was felt that it was better to have access to the gap condition directly, so that a base variable was available for direct manipulation, rather than to rely on the motor voltage signal.

Since access to this variable is a requirement for all the programs developed, its acquisition and processing is discussed here, whereafter the different individual control strategies are explored.

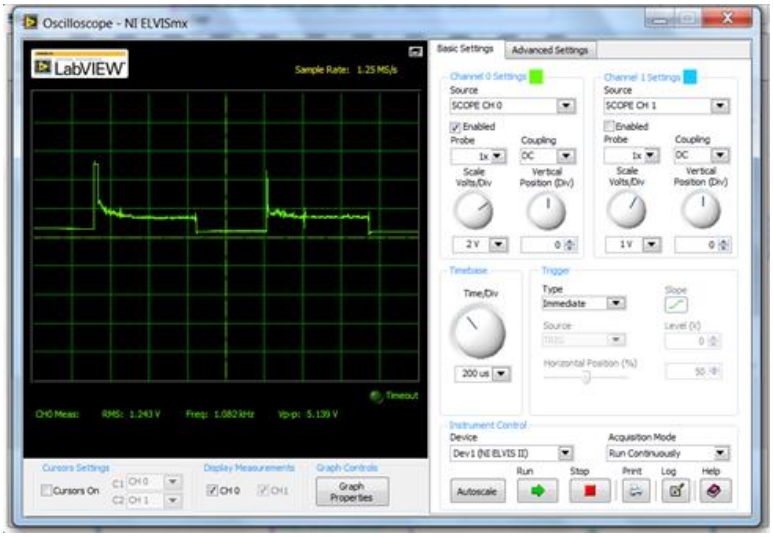
If the gap voltage signal used by Alic-1 itself had been readily available for extraction from the machine's controller, this would naturally have been used. However, since the Alic-1 circuit diagrams were not clear, it was difficult to see exactly where this signal resided, and even then, it would still have to be found in the physical machine, probably on a printed circuit board. Thus a technique where the gap condition was established directly from gap-based measurements was needed (the electrode and workpiece are both physically available for measurements to be taken from). Also, from an integration of EDM onto a CNC machine point of view, one does not want to have to rely on signals from the EDM machine. One wants a solution which could be applied potentially to any EDM power-pack – and if the machine's own signal is to be used, it would presumably need to be accessed differently for different machines, thus making the solution less generic.

6.5.1 Acquisition and Processing Strategies

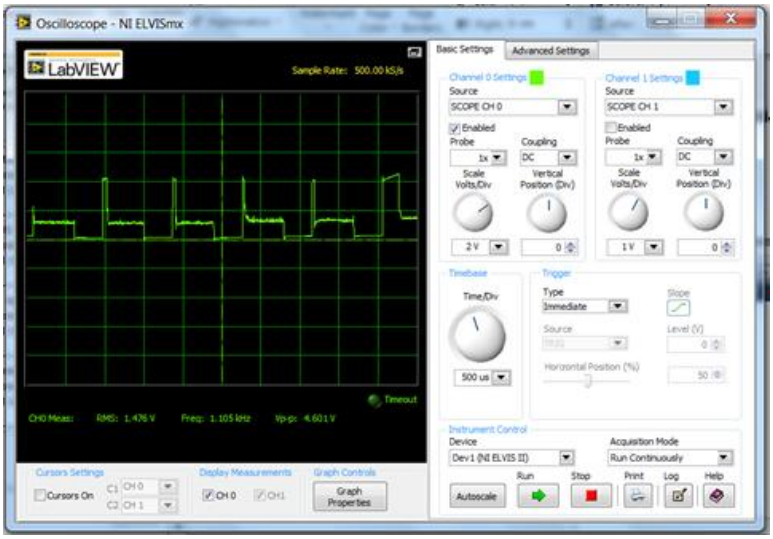
A number of techniques are commonly used for gap condition detection, as mentioned previously. It appears that the Alic-1 uses (some form of) average voltage as a gap condition indicator, and it was decided to implement a voltage based technique for use with the LabVIEW programs.

Before settling on a simple average voltage to indicate gap condition, some other voltage based options were considered. One problem with some techniques is that when long Off-times are specified for the EDM pulses, the long Off time significantly affects the average voltage value measured (causes it to drop) – whereas one does not want it to

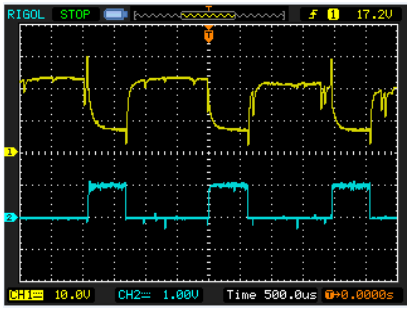
be affected by Off times, as the (instantaneous) gap voltage during this time is not an indicator of gap condition. The machine developed by Ben Fleming in his 'Build a Pulse EDM' book mentioned above suffered to some degree from this effect, although steps had been taken to limit it. The typical voltage profile during the EDM process (of a Pulse type EDM machine) is as shown in Figure 6.5 below. The profiles shown were captured using the LabVIEW Virtual Oscilloscope function (for ELVIS board), while performing EDM using the Alic-1 with its own servo motor, and using the gap detection circuit discussed below. Capture (b) shows some (small) variation in the 'On-delay' time (width of the narrow vertical spike). Capture (c) shows the gap current (Turquoise) coinciding with the voltage pulses. In (c), it appears that the gap voltage rises during the Off time – but this is a phantom voltage appearing on the oscilloscope.



(a)



(b)



(c)

Figure 6.5 EDM Pulse Profiles: (a) & (b): Voltage only; (c): Voltage and Current

It should however be noted that consecutive pulses are in general not very consistent in an EDM process. As the electrode moves (oscillates), and the gap distance changes, pulses look slightly different – even disappearing completely when gap distance is beyond a certain range. The electrode's actuator is not able to respond to each individual pulse (due to its inertia), so some averaging effect would be present anyway, even without proper filtering and signal processing. Often bursts of 'good' pulses occur (where gap distance is ideal), followed by a number of 'bad' pulses where over- or undershooting is occurring, or debris is affecting the gap distance.

It can be seen that after the initial turning on of the MOSFET, which places the spark generator voltage across the internal resistors and the gap itself, the gap voltage first rises to the generator voltage (since no current is flowing in the internal resistor, no voltage drop occurs across it). After a certain delay time (the On-delay time or Ignition delay time, which is gap distance dependent), the gap voltage drops as the arc strikes and the current starts to flow producing a voltage drop across the internal resistors. Burn voltage is typically fairly constant at about 40V. When the MOSFET Gate driving signal goes low, the MOSFET turns Off, the current is forced to stop flowing i.e. drops to zero, and the gap voltage drops to zero since there is effectively an open circuit at the MOSFET (between its Source and Drain). If the gap distance becomes affected during the burn (e.g. due to sudden debris formation or ejection), current may be forced to stop flowing, even while the MOSFET is still On. Often no arc forms during the On time, due to gap conditions not being ideal. In this case the gap voltage rises to the spark generator voltage when the MOSFET is On, and remains there during the whole On time. (This is a good thing, in that a larger average voltage means the electrode is too far away, and the control system will respond correctly to drive the electrode closer to the workpiece.)

It was at one stage thought that the voltage across the gap during the time of current flowing (the 'burn' time) was the important variable, and attention was then focused on obtaining knowledge of this voltage, for control purposes. Later it was found that the burn voltage did not vary much regardless of gap voltage setpoint selected on the Alic-1 machine, and thus was not really a good indicator of gap condition. Instead, it appears that the longer On-delay time occurring with larger gap distances causes the average voltage across the gap to be higher, because the gap voltage is high (generator voltage) during this time (i.e. before the arc strikes). Thus, to some degree, when one measures

and uses the average gap voltage signal, one is to a large degree really measuring the effect of the On-delay time.

During short circuits, i.e. when the electrode is actually touching the workpiece, the voltage across the gap remains low even when the MOSFET is On, as there can be no voltage developed across a complete short circuit (V=IR, and R is essentially zero). Some EDM machines also prevent the MOSFET from firing during periods when short circuits are occurring i.e. if it detects too many short circuits in a succession. Either way, this means that the average gap voltage is low when the electrode is too close to the workpiece.

In an attempt to exclude (long) Off times from affecting the measured gap voltage, a sample-and-hold (or rather a track-and-hold) approach was considered, either using a circuit built from appropriate components (capacitors, diodes, etc.), or using an off-the-shelf track-and-hold IC. (This function could also be achieved in LabVIEW.) The key to implementing a track-and-hold circuit is to know when the MOSFET in On, so that one can sample/track only during this time, and hold the value determined during this On time, during the Off time. In this way the Off time would not really affect the measured average gap voltage (the voltage of a number of consecutive EDM pulse cycle times would be used for averaging (perhaps using a 'moving window' approach); or some other averaging/filtering technique could be employed).

To detect On time means essentially detecting when the MOSFET Gate is On. This signal could perhaps be found in the Alic-1's electronic circuits; but it was preferred to use access points more readily available, i.e. external to the machine. A plan devised was to use the gap voltage itself, in combination with gap current, to detect On time, as follows (see Figure 6.6 overleaf).

If a voltage divider resistor pair was placed across the gap i.e. connected across the electrode and the workpiece, then when a voltage was detected across the (smaller) resistor (whose voltage would be proportional to the gap voltage), that would indicate that the MOSFET is On (as when it is Off the MOSFET resistance becomes essentially infinite as previously mentioned, so no voltage would be detected across the gap (or either of the voltage divider resistors), as all the generator voltage would appear across the MOSFET's 'open circuit'). However, it would be possible that the MOSFET is On,

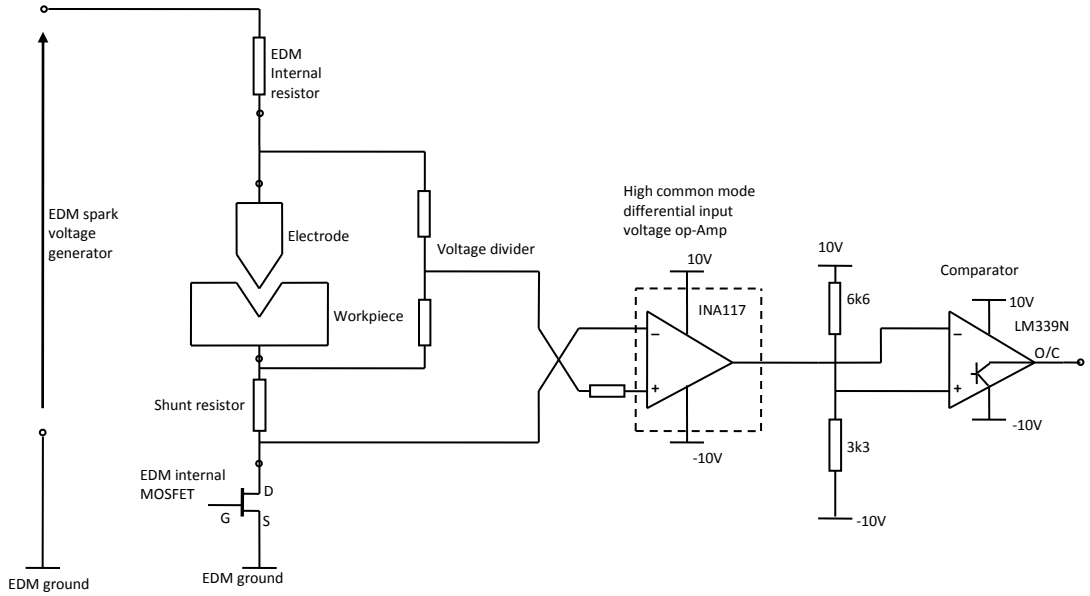


Figure 6.6 Circuit for EDM On-Time Detection

but a short circuit is occurring, in which case no voltage would be measured by the divider. To detect these occurrences, a small shunt resistor could be placed in series with the gap, such that, if voltage is detected across it, it means that the MOSFET must be On since a voltage across it would imply a current through it which would imply a current through the MOSFET. During Off time there would be no voltage across the shunt resistor; during On time but where open circuits were occurring (electrode too far from workpiece) there would also be no voltage across it (since no current is flowing through the gap). Thus it can be seen that neither the voltage divider nor the shunt resistor can be used in isolation to determine whether the MOSFET is On; but together they can detect all instances when the MOSFET is on. I.e. if there is a voltage across the voltage divider, OR if there is a voltage across the shunt resistor, then the MOSFET must be On. So these two voltages can evidence the MOSFET On state, using a logic OR function.

However, by probing across the gap in the method shown above, if a voltage was detected, it would imply that the MOSFET was On, as this effectively produces the logic OR function mentioned (voltage across either the voltage divider resistor, or the shunt resistor, will produce a voltage across the probing points). Thus a simple comparator could be used to detect the On state, by comparing the voltage measured across the probing points to zero (or rather a small voltage value), which would force the

comparator High when the probed voltage was higher than the (approximately zero) reference voltage.

The actual gap voltage would be measured across the smaller resistor of the voltage divider, but would be sampled during the On period and held during the Off. One problem remains, however, and that is that when the On detection circuit senses that the MOSFET has turned Off, if the track-and-hold function continues to track throughout the whole On time, and then holds during the Off, whatever value was at the track-and-hold input at the instant the On signal changed to Off, would (approximately) be held during the Off time. And this may in fact be a low or even zero voltage and not the burn voltage, as the gap voltage is busy dropping to zero at the same time as the detection circuit realizes that the MOSFET is turning (or has turned) Off. Thus one wants to force the 'hold' start time to be before the MOSFET turns off. One option is to make this 'hold start' time at a specific, fixed time after the MOSFT turns On. However, if a large On-time range is available on the particular EDM machine being used, as is often the case, then the 'track' time between MOSFET On, and 'hold start' would have to be made quite short so that it would not be longer than the shortest allowable On time that can be selected on the EDM machine. Nevertheless, this would still to a greater or lesser degree exclude the Off time from influencing the average gap voltage; and not too much accuracy would be lost as the voltage is fairly constant during burn time (once the arc has properly formed).

In the gap detection scheme envisaged, the 'track' start time would be dictated by the comparator output of the MOSFET On-detection circuit going High; thereafter a 'one-shot' timer would need to fire, and the expiration of that time would indicate the start of the 'hold' time. Note that the length of the 'one-shot' time could be made to be linked to the On-time selector knob on the EDM machine, so that longer track times are allowed when it is known that On-times will be longer.

The circuit of Figure 6.7 overleaf depicts an envisaged arrangement for detecting the EDM machine's MOSFET On-time (as previously), but in addition using this signal in a track and hold arrangement. The lower left op-amp measures the gap voltage using the voltage divider; at its output is an Optical-FET switch. When the comparator (top right) detects the MOSFET is On, it's open collector output would be pulled down, allowing

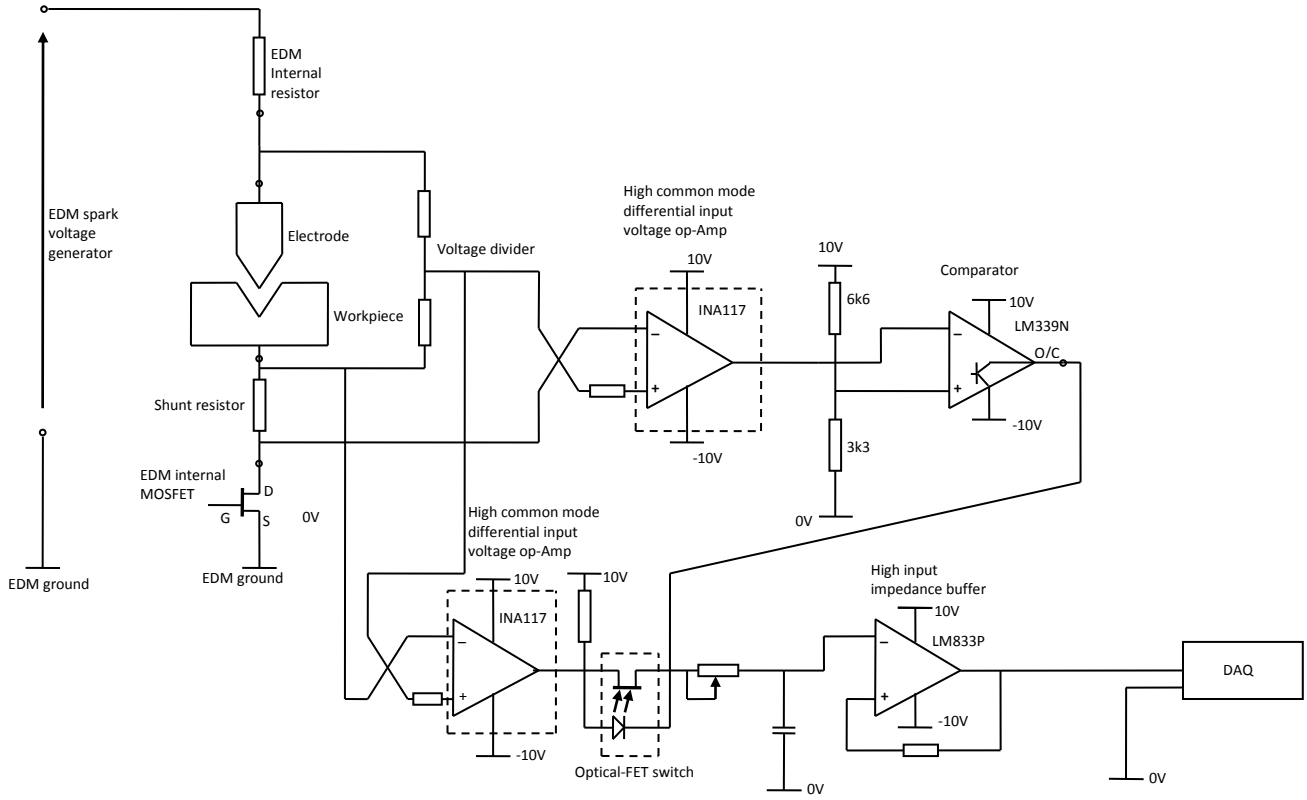


Figure 6.7 Circuit for EDM On-Time Detection using Track & Hold

current through the Optical-FET switch emitter, driving its 'Gate'. The FET switch would turn On and its resistance would go low, and the capacitor would charge. When the EDM MOSFET turns Off, the comparator open collector would not be pulled low, and no current would flow through the Optical FET switch's emitter; thus its resistance would go very high, such that the capacitor would not be able to discharge through the FET switch. In this way, the voltage present at or near the time when the EDM MOSFET turns Off would be maintained at the buffer op-amp's input, and so fed into the DAQ device. The buffer op-amp would need a very high input impedance, so as not to drain the capacitor during the Off-time (thus the LM833P would be suitable).

For simplicity, in the end just a voltage averaging (i.e. filtering process) was implemented (in LabVIEW), and the track-and-hold function was not actually used. Although long Off-times can influence (reduce) the average gap voltage, the more important parameters are, in a sense, the repeatability and stability (smoothness) of the gap measuring technique, since the gap voltage setpoint can be manually adjusted on the LabVIEW front panel such that desired EDM conditions are achieved (if the process is stable, and within a certain gap distance range, EDM will take place, but parameters such as surface finish and material removal rate will be affected).

6.5.2 Circuit for Gap Voltage Measurement

The gap voltage signal was measured using a voltage divider of suitable ratio (10:1), such that a voltage not higher than about 10V would occur across the sensing resistor (even when open circuits were continually occurring, i.e. the gap voltage itself would be about 100V, the voltage of the EDM generator). This was because a 10V (and -10V) power supply for the sensing circuit was to be used (and the DAQ device itself does not accept signals higher than 10V on its analogue inputs). The circuit implemented is depicted in Figure 6.8 below, and is discussed further.

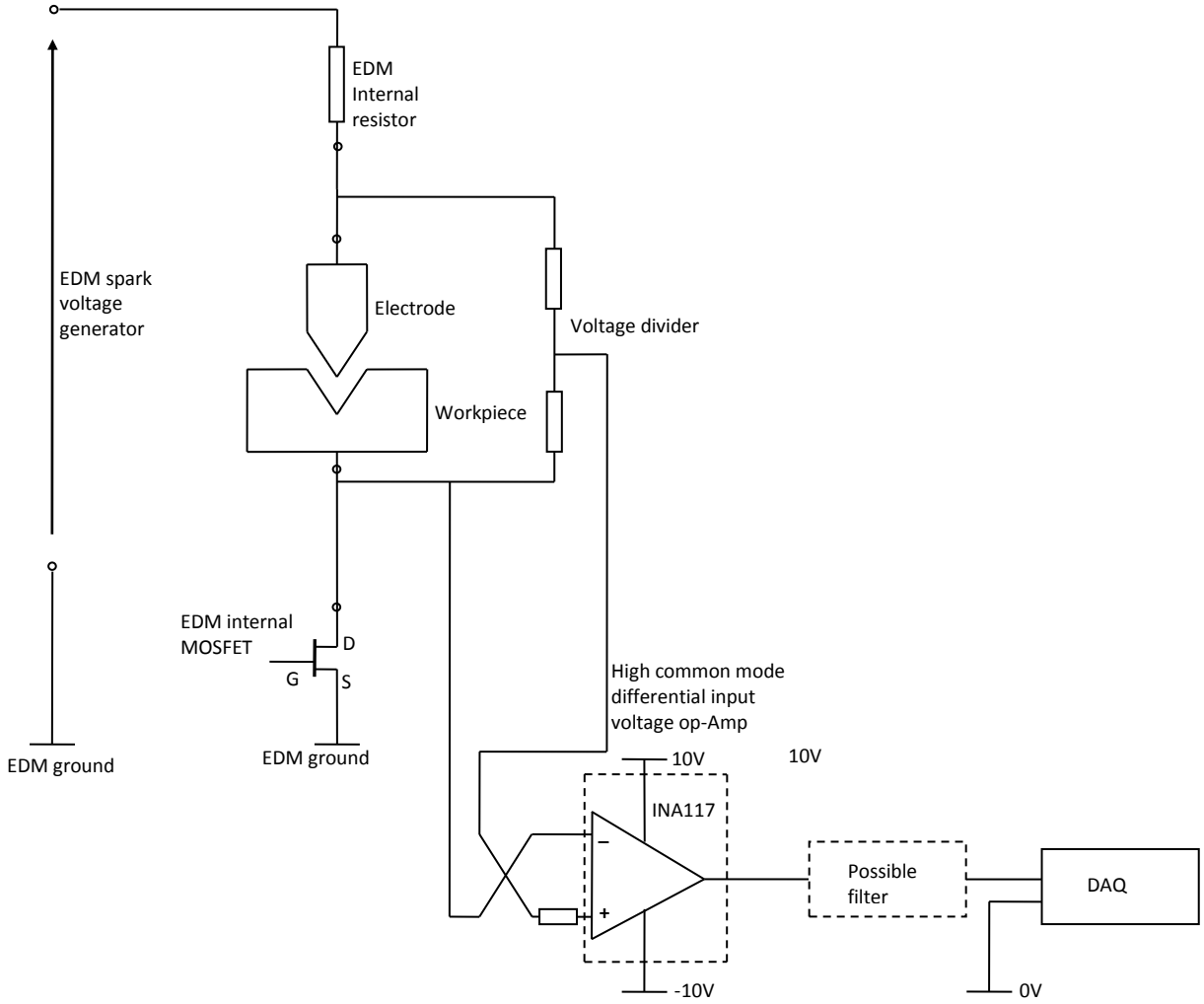


Figure 6.8 Circuit for Gap Voltage Measurement and Isolation

One concern in obtaining the gap voltage from the EDM machine is that the grounds of the EDM machine and the DAQ device (and PC and power supply for the gap sensing circuit) may not be at the same voltage (and one does not want to force them to be on

the same ground as the EDM process being extremely noisy with many voltage spikes, may cause additional noise on the gap sensing circuit). Thus a number of isolation options were considered, including using voltage to frequency conversion (and back to voltage), and a linear analogue opto-coupler. It was decided to implement an op-amp circuit utilizing a special high common mode, high input impedance, op-amp (the AD629 or INA117). The use of this op-amp does not create complete galvanic isolation as such, but does to a greater or lesser degree make the EDM gap 'side' of the op-amp independent of the power supply and output 'side', in terms of grounding and voltage levels. The op-amp has unity gain. When using this op-amp, a resistor of the same value as the sense resistor has to be used on the appropriate input, to balance the resistor network of the op-amp (the sense resistor being the lower resistor of the gap voltage divider pair, across which the differential voltage is being measured). This resistor can be seen in the schematic of the sensing circuit of Figure 6.8.

A Ferrite bead (toroid) was used on the leads connected to the gap (i.e. electrode and workpiece), physically just before they connected to the voltage divider. The leads were passed together through the coil a number of times, producing a number turns around the bead. A vast improvement in the gap signal in terms of noise (as seen on an oscilloscope) was visible when this bead was employed. The picture below shows the system during testing (the Alic-1 servo head is being used). Leads can be seen connected to the electrode and the workpiece. The resistor voltage divider circuit and the Ferrite bead are located on the prototyping board visible at the bottom right.

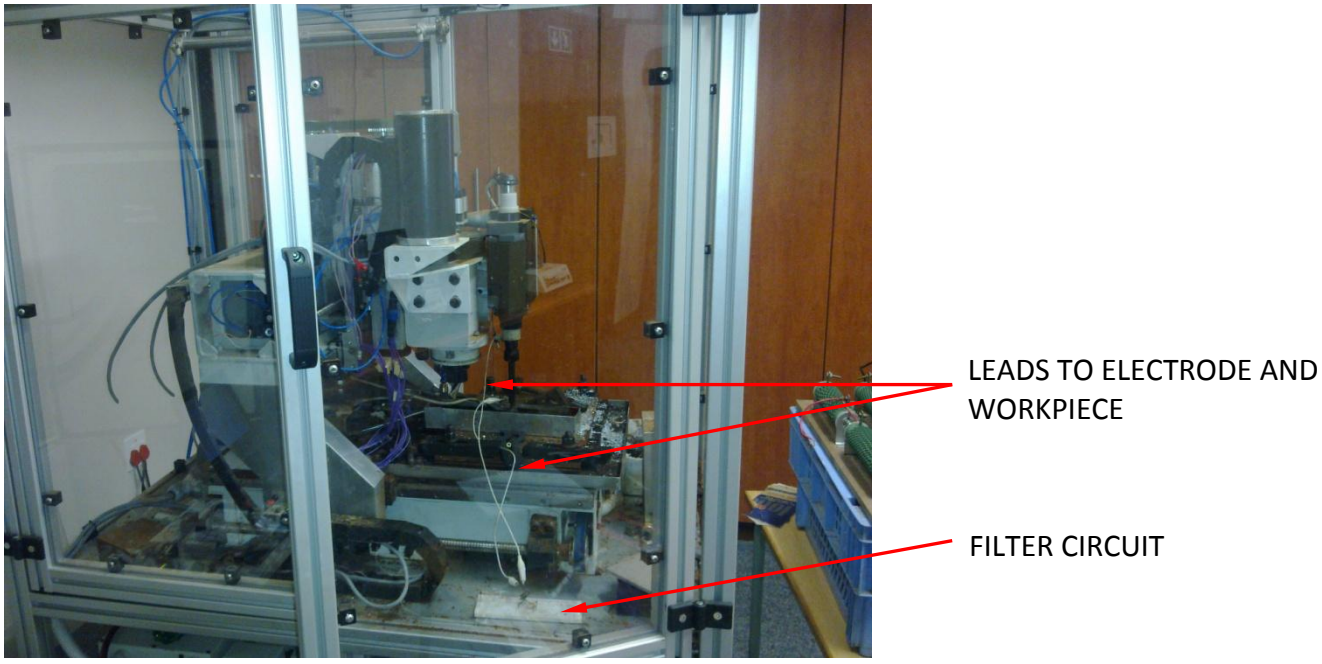
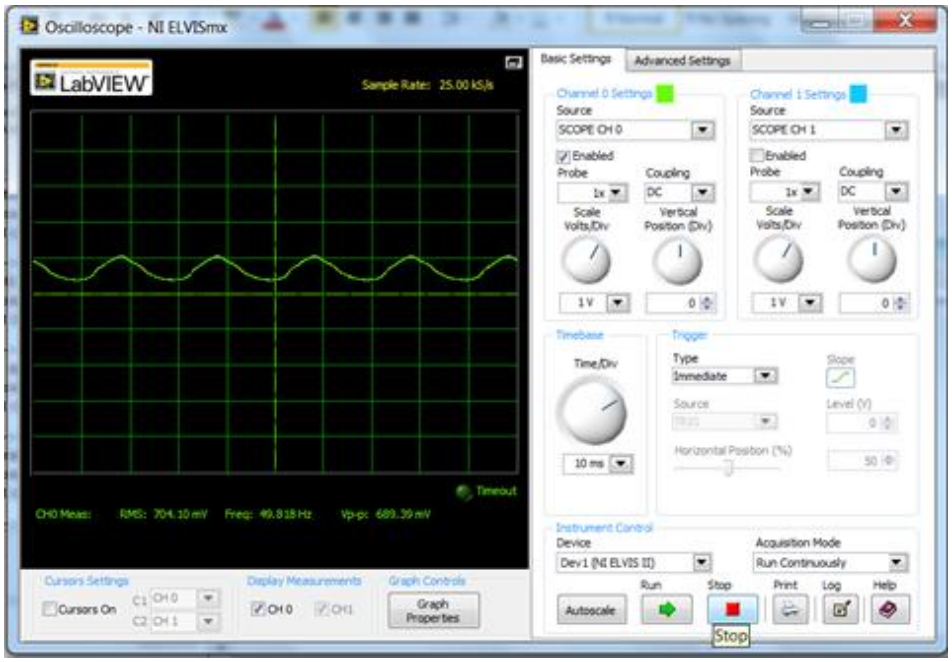


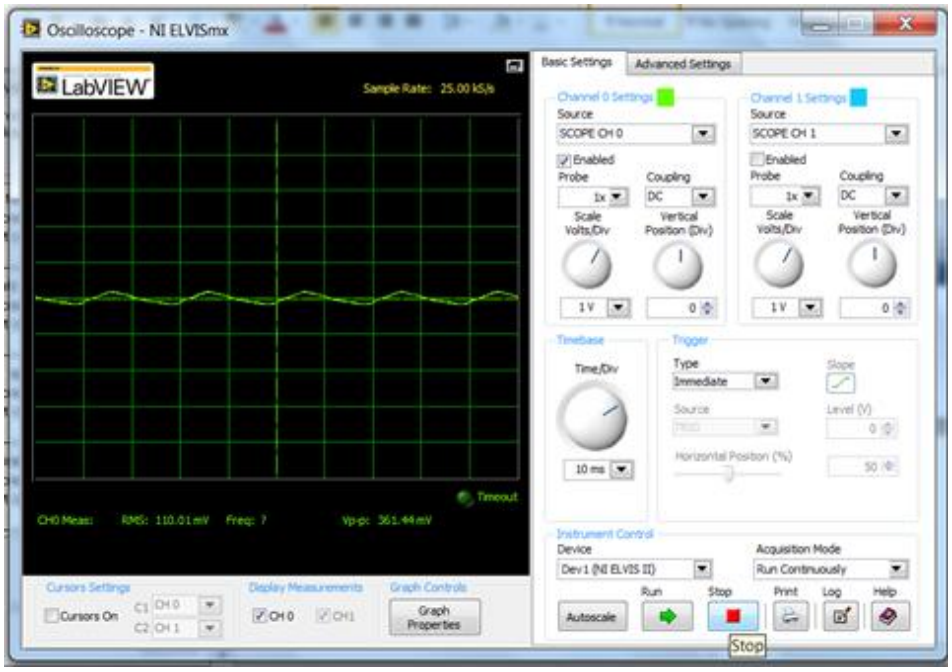
Figure 6.9 EDM Gap Voltage Filter Testing

6.5.3 Filter Implementation

The gap voltage itself is not a very readily usable signal. Firstly, it has the form of pulses, as seen previously. Secondly, these pulses are not of a consistent shape. Thirdly, a lot of EMI (Electromagnetic Induction) noise is present at the gap (and on leads connected to the gap, if not extremely well shielded), due to the 'spiky' sparking process. Fourthly there appeared to be 50Hz mains noise superimposed on the gap signal. A number of oscilloscope captures of gap voltage are shown in Figures 6-10 through 6-12 (below, overleaf, and on the page following), and are discussed further.

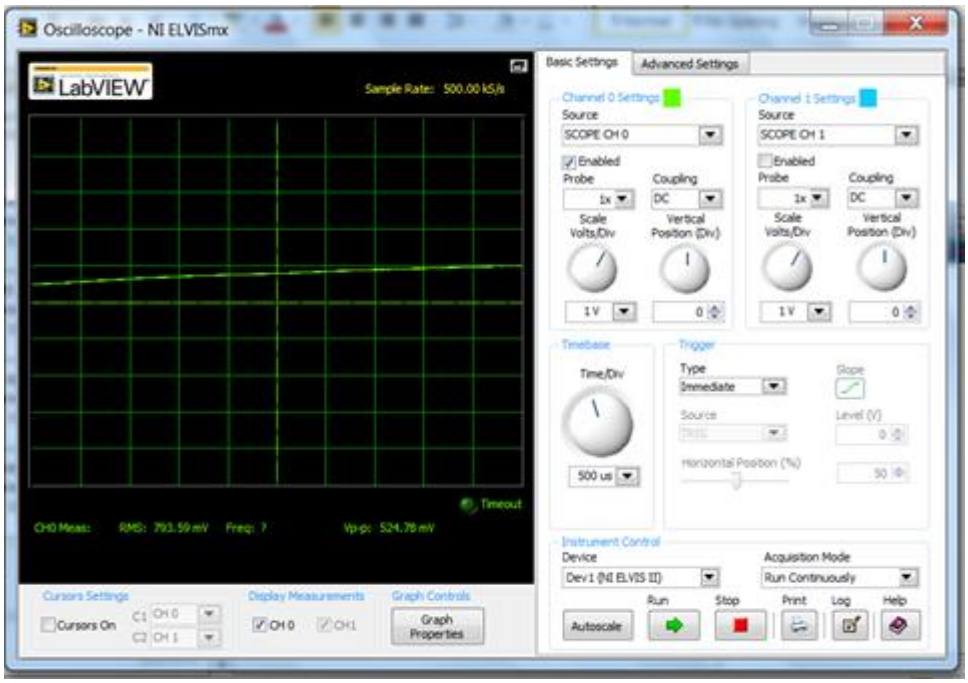


(a)

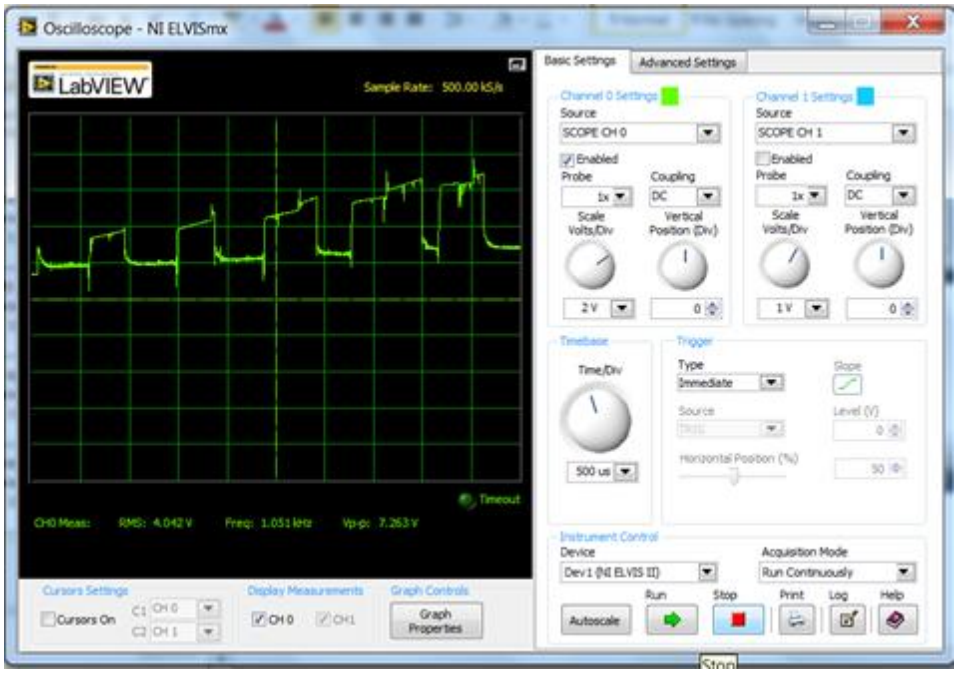


(b)

Figure 6.10 Gap Voltage with EDM machine (a) on Standby, and (b) Completely Off

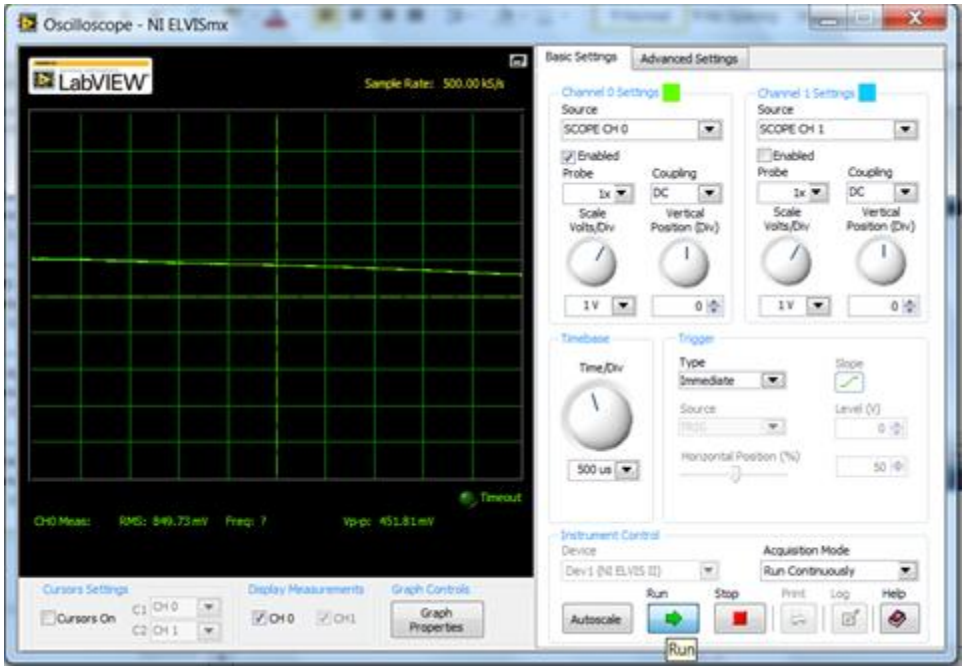


(a)

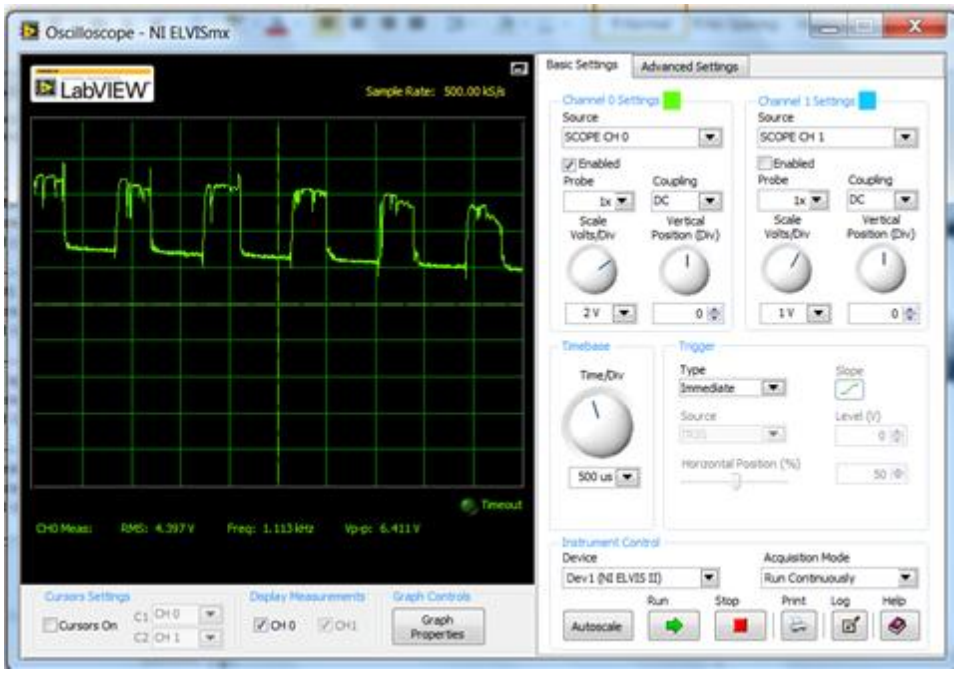


(b)

Figure 6.11 (a) Gap Voltage Noise at Faster Time Scale (Rising Voltage), and (b) EDM Pulses Superimposed on Rising Noise Signal



(a)



(b)

Figure 6.12 (a) Gap Voltage Noise at Faster Time Scale (Dropping Voltage), and (b) EDM Pulses Superimposed on Dropping Noise Signal

Some of the above captures display only the noise, present even while the EDM machine is off or on standby, while others display the EDM pulses superimposed on the noise signal, during sparking. In some instances 100Hz (first harmonic of mains frequency) is also visible. These frequencies appear to be the result of interference from sources outside the EDM machine, and in that sense are a form of EMI; but as can be

seen from the figures, there is a difference between when the machine was on standby, vs. when it was off completely – thus some of the mains noise appears to be produced in the machine itself, and is thus not EMI specifically. All the above points considered, filtering of the gap voltage signal is clearly necessary.

Filtering in hardware was considered (and was even partly implemented), but it appeared the most versatile method would be to use LabVIEW to provide suitable filtering of the gap signal. The filter developed in LabVIEW is shown in Figure 6.13 below.

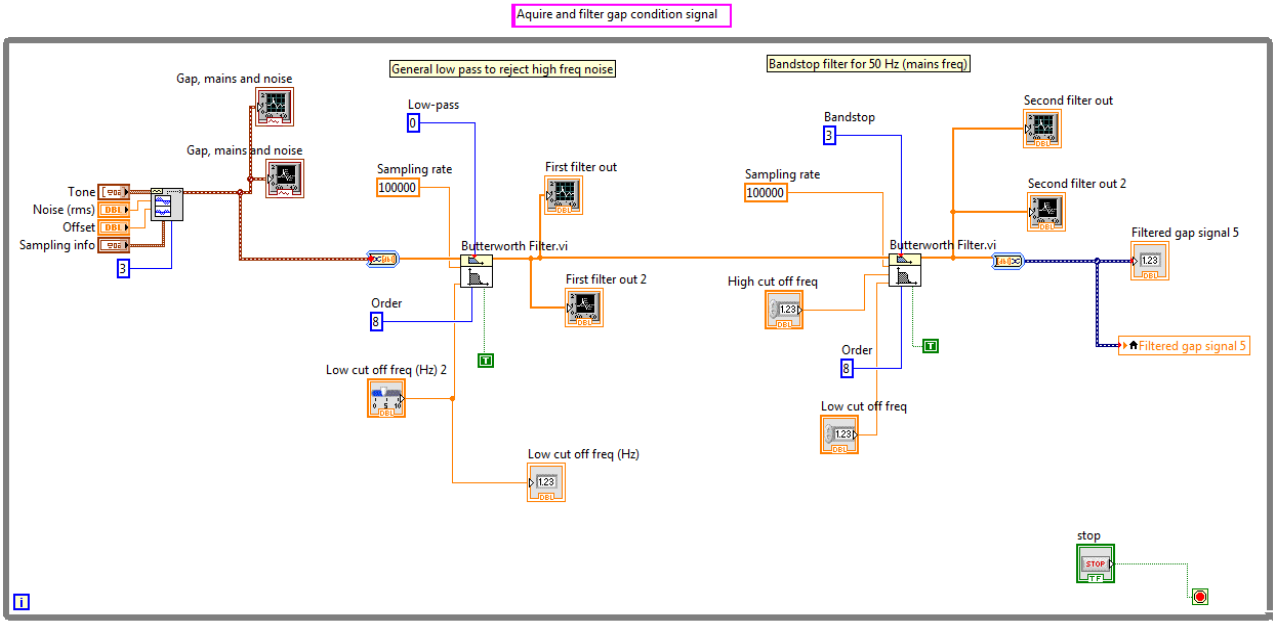


Figure 6.13 Filter Implementation in LabVIEW (Two Filter Version)

The digital implementations of a number of types of common filters are available in LabVIEW, e.g. Butterworth, Bessel, etc. Butterworth filters were used in this instance. A version with three Butterworth filters, arranged in series, was developed (a two filter version is shown in the diagram above, for simplicity). The EDM gap signal is represented for testing purposes as a Tone & Noise generator VI function block. This enabled specific superimposed frequencies and noise to be specified to represent the gap signal (set on the front panel). A DC offset was included (selectable in the Tone function), such that the overall waveform would be more similar to real EDM gap voltage signal, which is always positive.

The first filter is a low-pass filter, to remove all the high frequency noise from the gap signal. This noise is related to the pulse frequency of the EDM process itself (typically in

the range of 1 kHz to 100 kHz, because pulse periods, i.e. combination of On and Off time typically range from 1000 microseconds to about 10 microseconds), harmonics induced by the pulsing process, and potentially aliasing frequencies induced by DAQ sampling frequencies, if they are too low relative to frequencies present in the gap signal (as per the Nyquist criterion, which states that sampling frequencies should be at least twice the frequency of the highest frequencies occurring in the signal, if the signal is to be properly portrayed; but using high sampling frequencies could put load on the processor, unnecessarily). In some experimentation however, purposefully chosen (relatively) low sampling frequencies did exclude some high frequency noise of the signal itself (aliasing frequencies induced would be less than these).

The second and third filters (only second included in version shown above) were used to remove the specific frequencies of noise (namely 50 Hz and 100Hz), that had been observed on the gap signal from the EDM machine, even when sparking was not taking place. Thus they are band-stop filters, arranged as high order notch filters (attenuation is still 'rolling off' from each side of the 50Hz or 100 Hz notch); corner frequencies were chosen appropriately, a little above and below each notch frequency.

The use of filters naturally produces an unwanted phase lag into the signal, especially if high order filters are used. Physical response of the actuator, due to its inertia, based on evidence from the initial testing of the Alic-1 and from other sources, and bearing in mind that the linear actuator has a significantly lower mass than the moving parts of many other EDM machines, was not expected to be higher than about 70Hz (thus it could be argued that the 100Hz notch filter and the general low-pass filter are not really necessary, although there are reasons for their use as explained later). Also the sampling required for each filter puts additional load on the PC's processor, which may adversely affect the update time of the (more important) Do loops of the main program. Filter sampling rates of the filters themselves had to be fairly high, so as not introduce aliasing frequencies themselves. Although the inertia of the actuator/electrode would limit response to higher gap signal frequencies, they are still undesired from a PID smoothness point of view (the derivative component, in particular, would be significantly affected); also they cause many, many sign (polarity) changes of the gap signal. This can be problematic for the control system version utilizing shift registers as discussed in section 6.7 – the polarity changes cause many, many changes in shift register

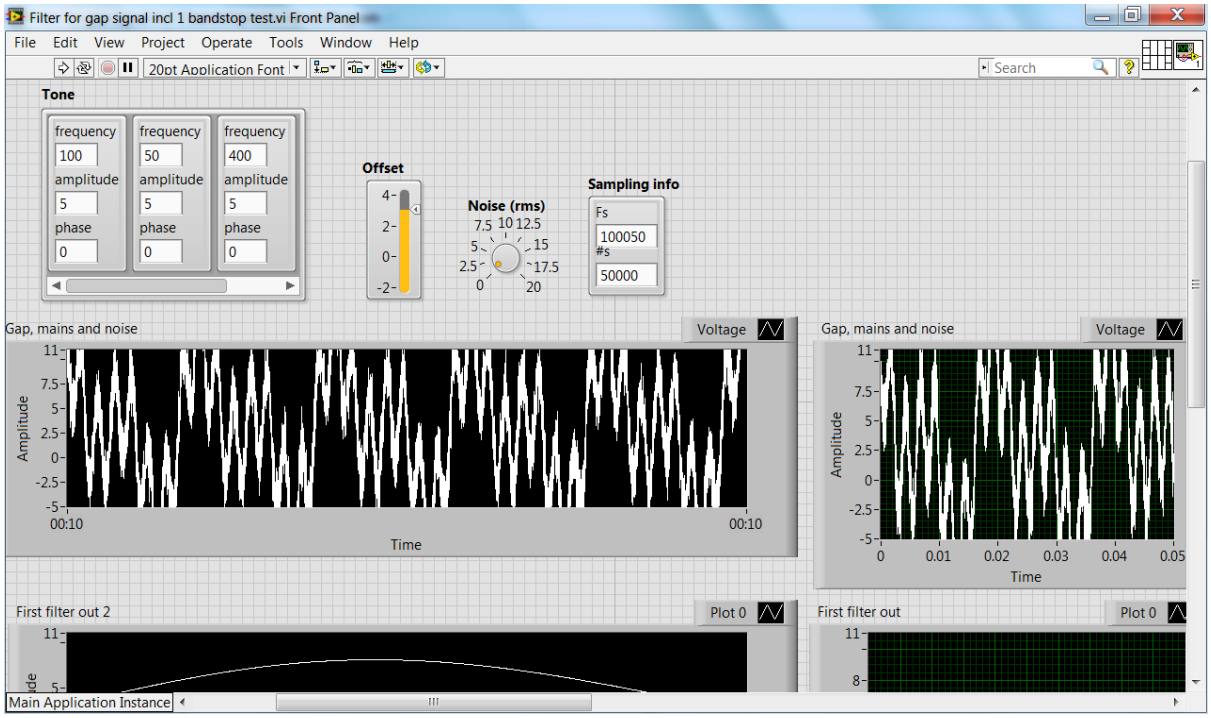
incrementing vs. decrementing modes, which is particularly unwanted, as explained later.

It was found in experimentation that the second notch filter (100Hz) made the gap signal if anything slightly less stable/smooth than if only one notch filter was used, and it was thus omitted in testing, i.e. removed from the LabVIEW filter Do loop.

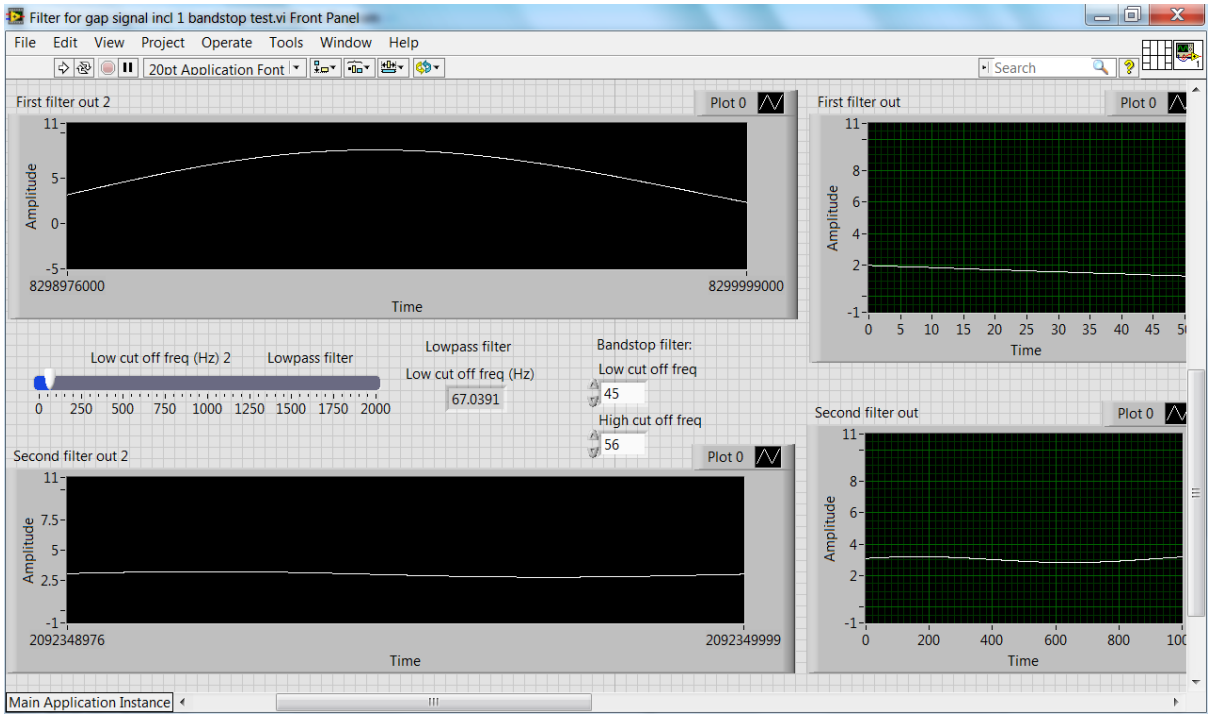
Filter performance was evaluated, visually by using LabVIEW front panel graphs for the unfiltered signal, and after each filter. In particular frequencies of 50 and 100 Hz were chosen, to check the filter's ability to reject these frequencies (mains noise and first harmonic). The cut-off frequencies for each filter were made adjustable by pointer sliders on the front panel so that response at different chosen frequencies could be easily evaluated. The general low-pass filter cut-off frequency was usually chosen at about 200Hz, thus excluding high frequency noise but still allowing potential response to frequencies a little higher than the maximum oscillation frequency expected from the actuator. It was naturally found that cut-off (3db corner) frequencies for the notch filters had to be significantly lower and higher than the notch frequency desired to be excluded (e.g. 40 and 60 Hz, to exclude 50 Hz), in order to significantly exclude the notch frequency. High frequency noise was rejected well by the filter.

Results of the testing of the filter arrangement are shown in Figures 6.14 through 6.17 overleaf. In each figure the settings are different, for testing purposes. Explanations and interpretations of each scenario are given. The particular filter version being tested here has only two filters (one lowpass and one bandstop).

In Figure 6.14 overleaf, the noise setting is minimal. The cutoff frequency (seen in (b)) for the lowpass filter is set quite low, at 67Hz. In (b) one can see that a 50Hz signal has propagated (unwanted) through the filter, but it has been attenuated by the second filter (50Hz notch filter), and thus the signal in 'Second Filter Out' is usable. (Any high frequency noise, and the 100Hz signal, have already been attenuated by the first filter (lowpass).)



(a) Front Panel Showing Tones, Noise, and DC Offset Settings, and Combined Waveform

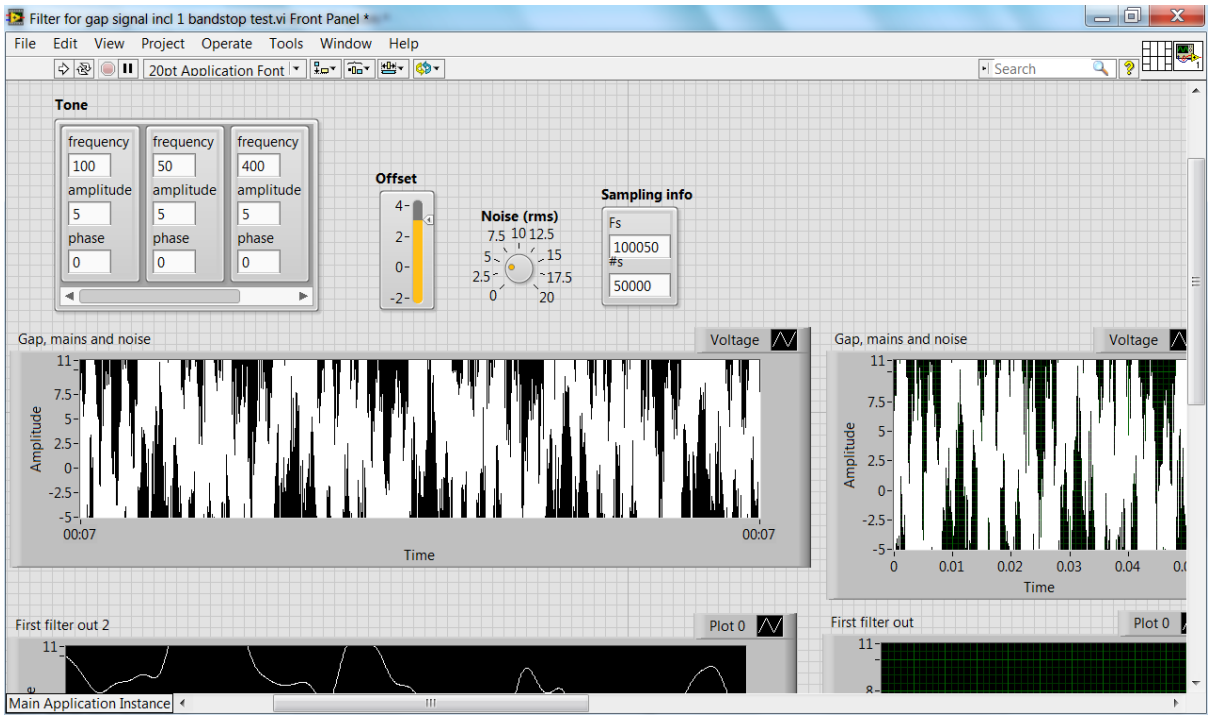


(b) Front Panel Showing Waveform Present after Lowpass Filter (upper), and after Bandstop Filter (lower)

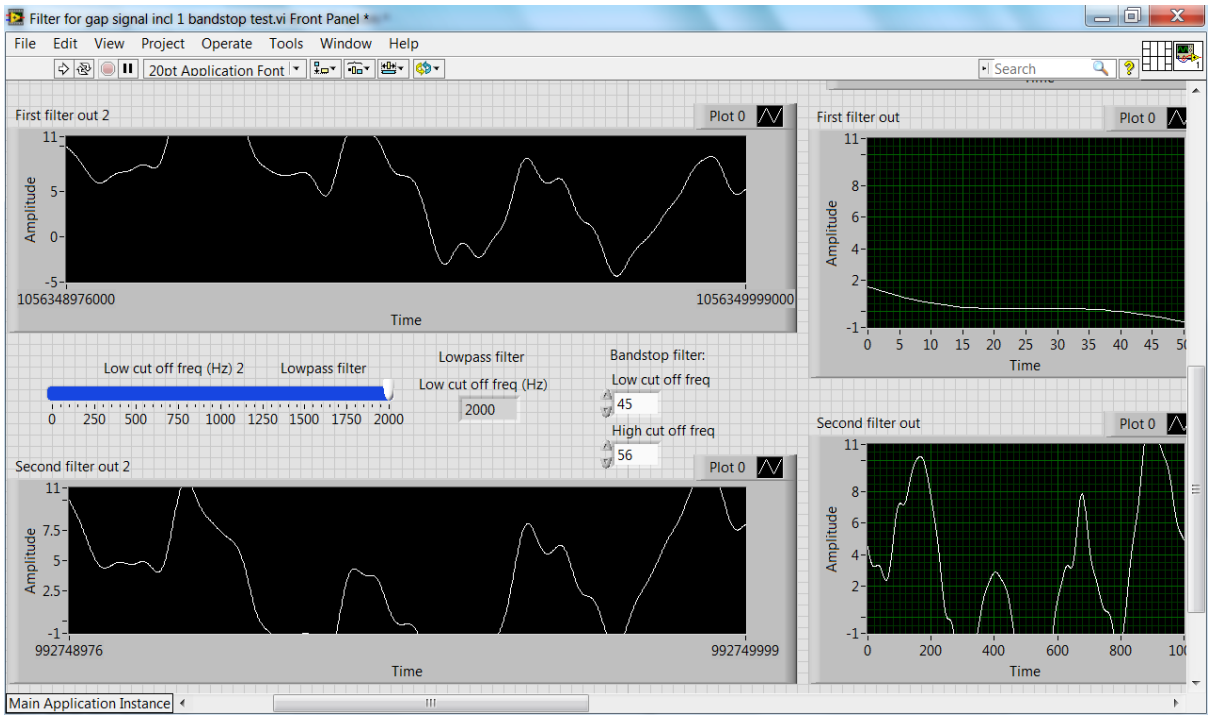
Figure 6.14 Filter Performance Testing (Tones, Noise, and DC Offset), Scenario 1

In Figure 6.15 overleaf, the Noise amplitude has been increased significantly, and the cutoff frequency for the lowpass filter has been set very high, at 2000Hz. All frequencies except the high frequency noise have propagated through the first filter (lowpass). The

second filter, being only a notch filter, is unable to attenuate the 400Hz and 100Hz signals, and thus the output is not usable.

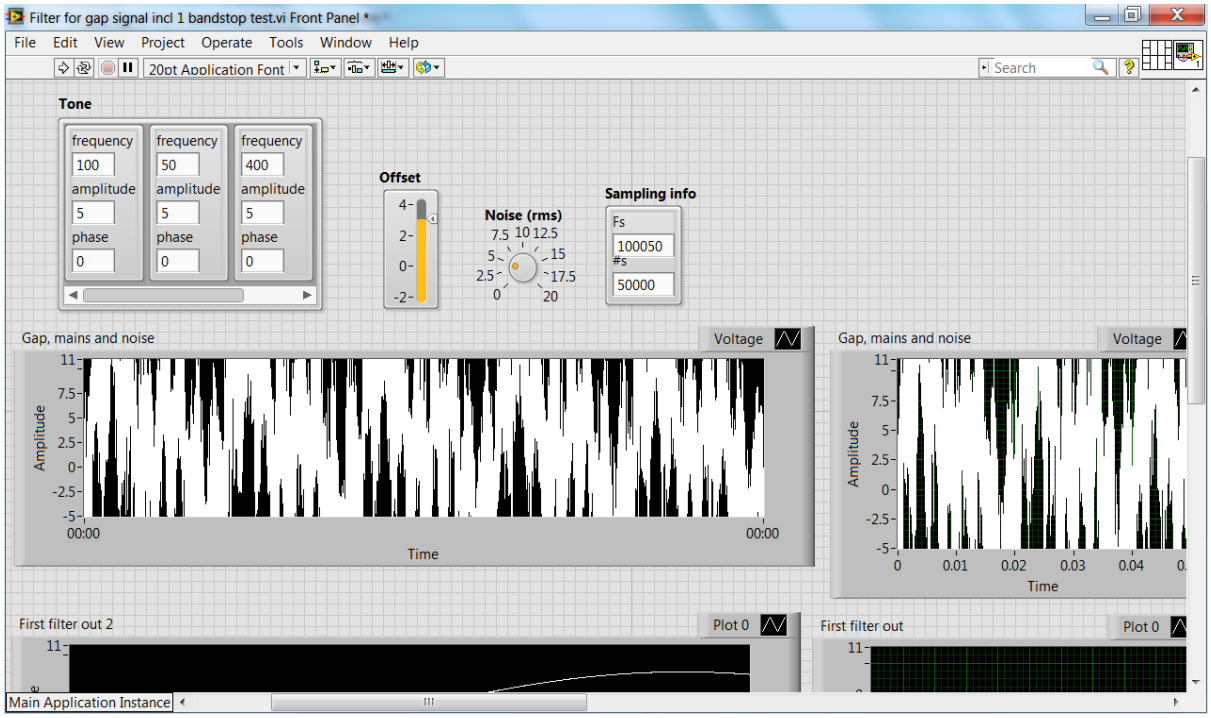


(a) Front Panel Showing Tones, Noise, and DC Offset Settings, and Combined Waveform

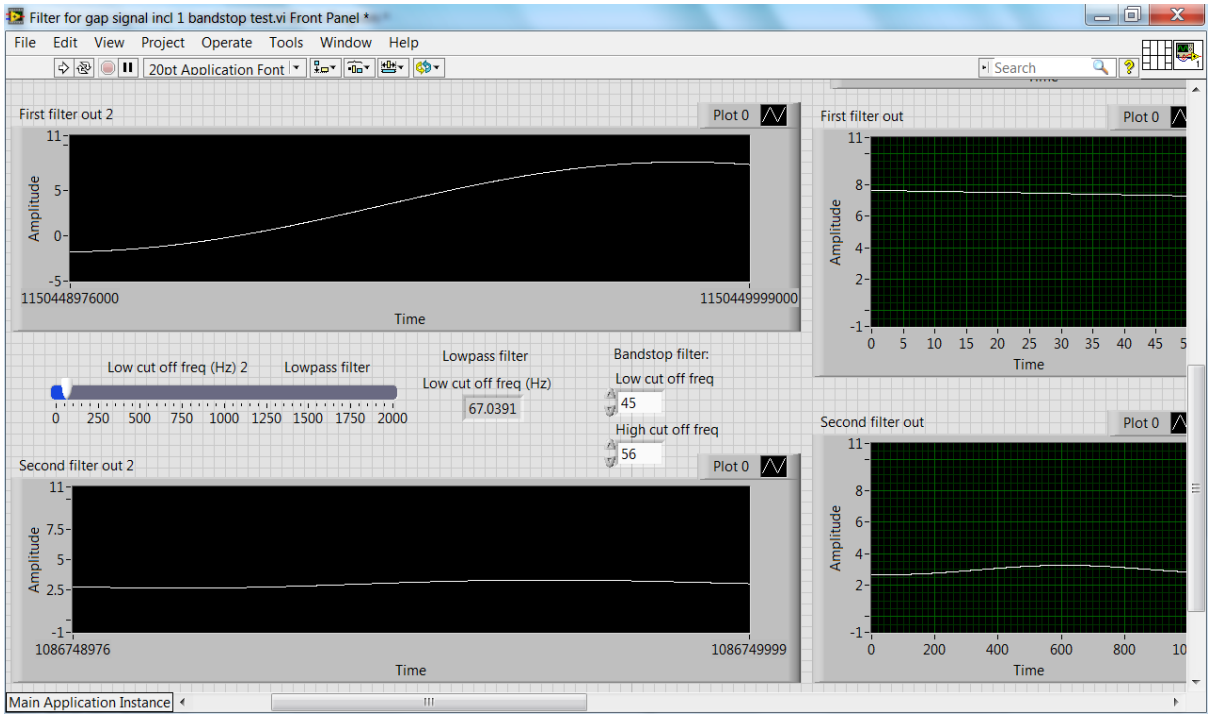


(b) Front Panel Showing Waveform Present after Lowpass Filter (upper), and after Bandstop Filter (lower)

Figure 6.15 Filter Performance Testing (Tones, Noise, and DC Offset), Scenario 2



(a) Front Panel Showing Tones, Noise, and DC Offset Settings, and Combined Waveform

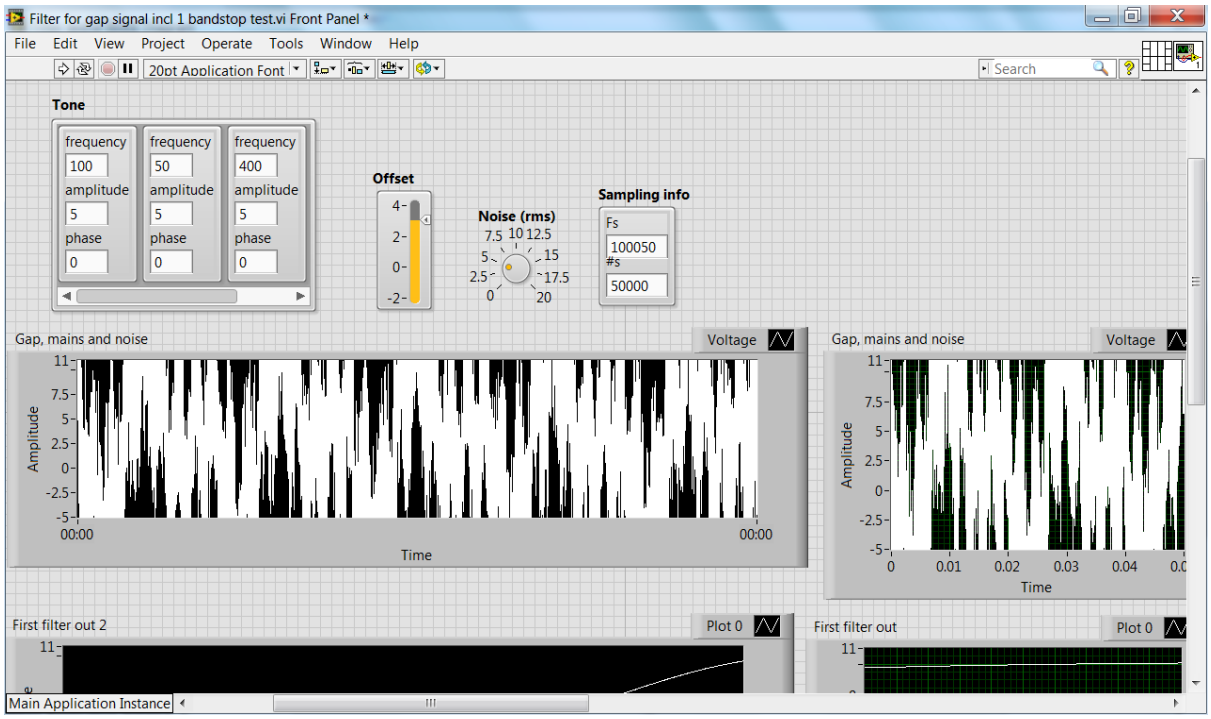


(b) Front Panel Showing Waveform Present after Lowpass Filter (upper), and after Bandstop Filter (lower)

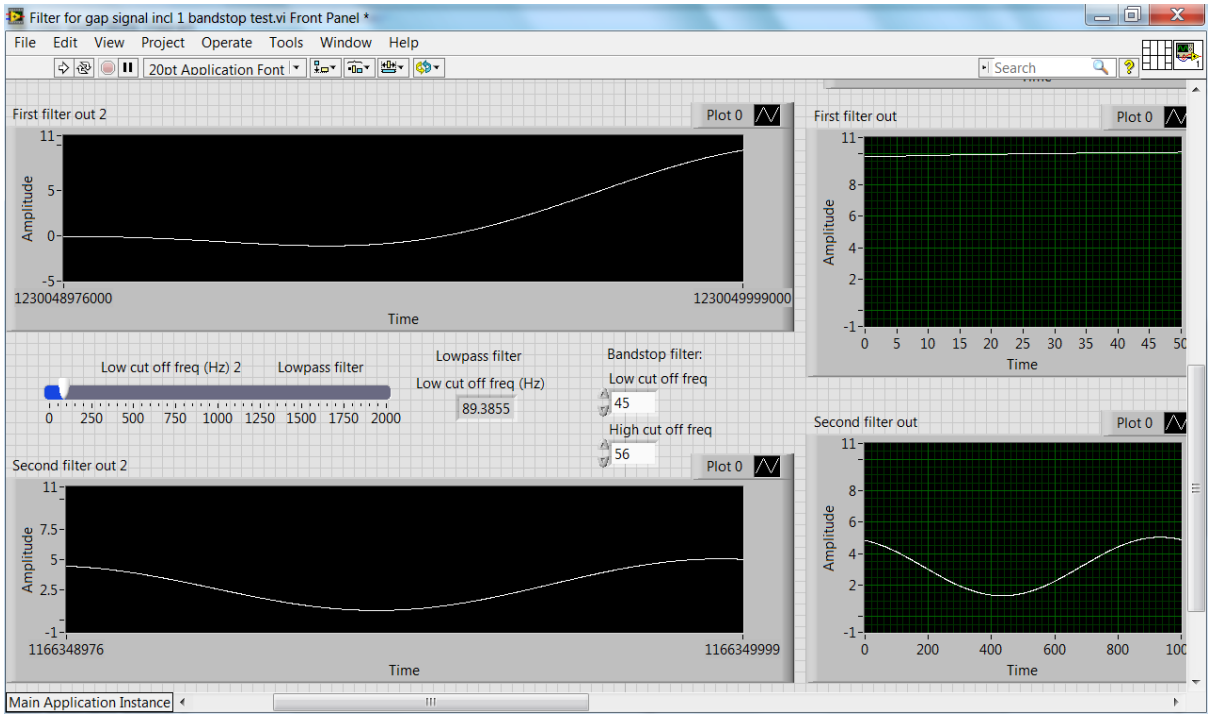
Figure 6.16 Filter Performance Testing (Tones, Noise, and DC Offset), Scenario 3

In Figure 6.16 above the Noise setting is still high, but the lowpass filter's cutoff frequency has been set low again, at 67Hz. Here the noise and the 400Hz signal and

the 100Hz signal have been attenuated, by the lowpass filter, and the 50Hz signal has been attenuated by the 50Hz notch filter; thus the ‘Second Filter Out’ signal is usable.



(g) Front Panel Showing Tones, Noise, and DC Offset Settings, and Combined Waveform



(h) Front Panel Showing Waveform Present after Lowpass Filter (upper), and after Bandstop Filter (lower)

Figure 6.17 Filter Performance Testing (Tones, Noise, and DC Offset), Scenario 4

In Figure 6.17 on the previous page, the lowpass filter's cutoff frequency has been raised to 89 Hz, and the 100Hz signal has only been partly attenuated. The second filter has attenuated the 50Hz signal, but some of the 100Hz signal has still propagated through the combination as the notch filter is unable to address the 100Hz signal which has (partly) passed through the first filter.

The results of testing of a version of the LabVIEW filter using the real gap signal from the EDM machine itself (Alic-1) are shown in Figure 6.18 below. The typical gap signal profile can be seen, but the filtered signal appears quite smooth thus the result is good. The lowpass filter cut-off frequency can be seen set at just over 100Hz.

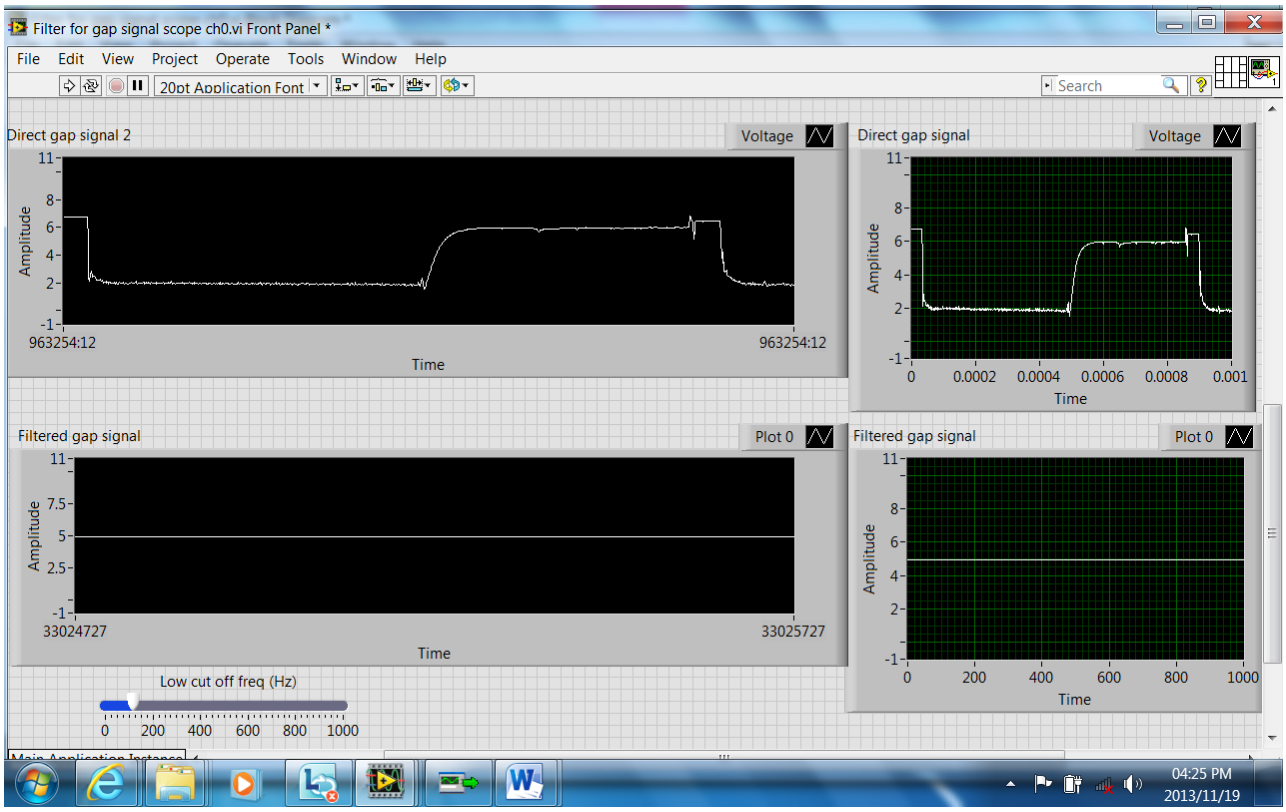


Figure 6.18 Filter Performance with EDM Machine Real Gap Signal

Testing (in a separate testing program, with no notch filter) also used a square wave generator (PWM generator, where duty cycle could be adjusted, and edges were sloped), with a Gaussian White Noise signal superimposed, to roughly represent the EDM pulses. The square wave was able to be converted to an almost DC representation, satisfactorily. The test program in shown in Figures 6.19 overleaf, and typical results in Figure 6.20 on the page following. On the front panel, the individual Noise, PWM, and Combined waveforms can be seen, along with the individual filtered

Noise, PWM, and Combined (Filtered Total) waveforms, such that one can see the individual components and how they contribute to the final filtered signal.

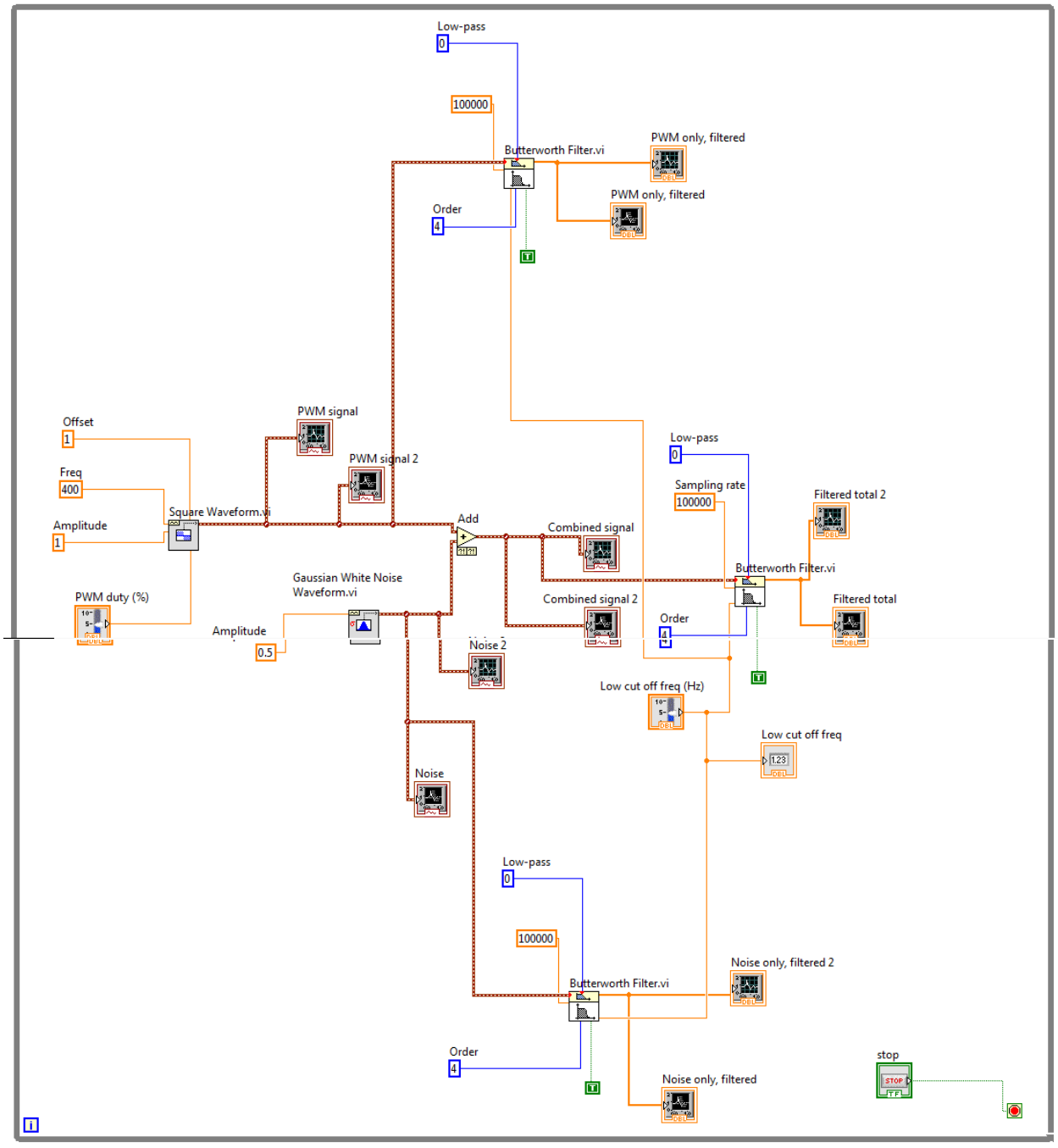


Figure 6.19 Filter Test Program using Square Wave and Gaussian White Noise

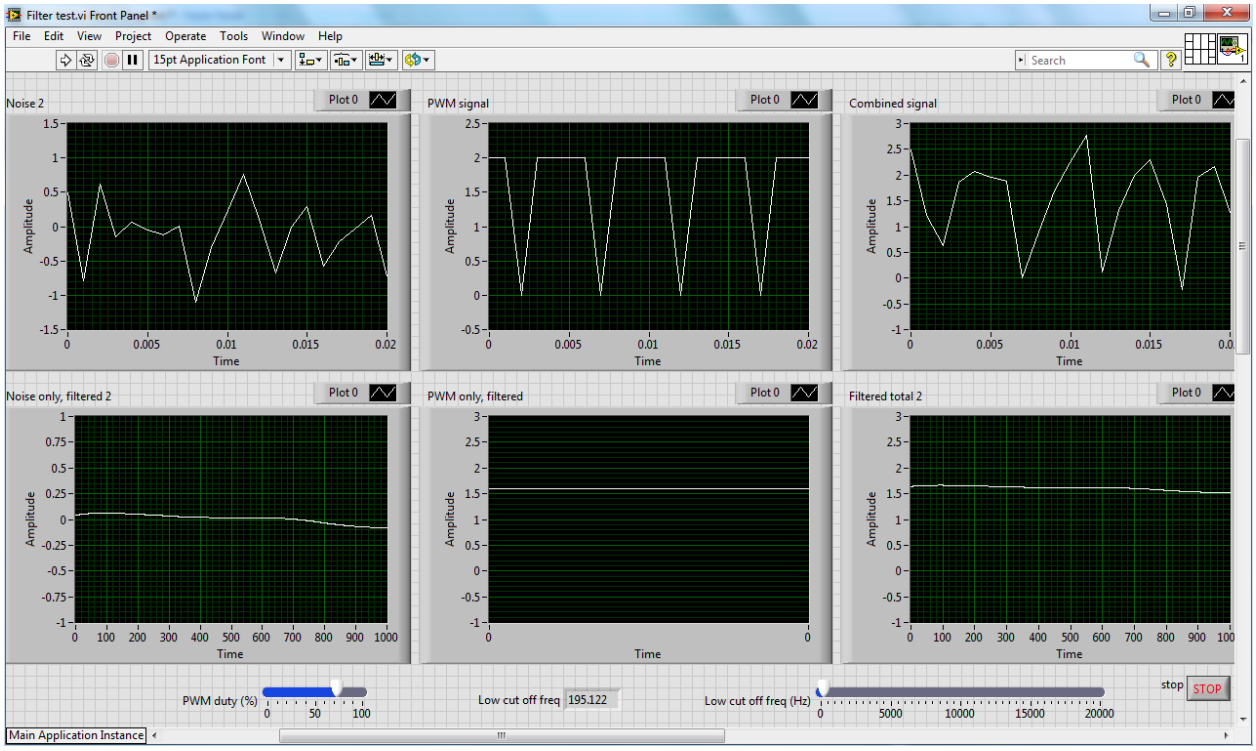


Figure 6.20 Filter Performance Testing (Square Wave and Noise)

Ideally the filtering program should run on a different device to the PC running the main control program, so that the main program processor is not loaded unnecessarily by the filtering program, which uses fairly high sampling frequencies. A suitable device for this would be, for example, a National Instruments Single Board RIO, a DAQ device having its own processor. Filtering results could be passed from the board to the main program via a Shared Variable, as the board, although running its own program, can still maintain communication with the main (host) PC running then main program. This as opposed to using an analogue output on the board for the filtered signal, and reading it in to an analogue port on the main DAQ device (the ELVIS development board in this case) – although this would still be better than performing all the filtering in the main program, as much lower sampling frequencies could be used on the analogue input port since high frequencies would have been excluded already.

6.6 Single Axis Control Programs

6.6.1 Simple Program, Single Loop

The simplest program developed is a single control loop where the input (actuating signal) is the EDM gap (electrode to workpiece) voltage and the output (controlled variable) is the varying PWM signal to the motor driver (or the output can be considered to be the motor voltage itself (which is proportional to the PWM duty cycle), depending on whether one is considering only the LabVIEW program itself, or the control system as a whole including the physical devices). Gap voltage is mentioned as the actuating variable, but this is really just a representation of gap distance, which is difficult to measure directly. Other representations could also be used, e.g. on-delay time as mentioned above. This control method would be useful only for single axis control (the system requirements for multi-axis control are discussed later).

A simplified control block diagram of this simple control system is shown in Figure 6.21 below. A more representative version is shown overleaf in Figure 6.22, where the physical devices providing the functions of the simplified diagram are shown; the diagram below that (Figure 6.23) is still more realistic, as the H-Bridge does not accept 'negative' signals. The LabVIEW portion of the system is indicated and forms the 'controller' function, a signal manipulation function (conversion to PWM), as well as the filtering function to produce a smooth value from the gap voltage pulses. Naturally there is only one DAQ device, the NI ELVIS development board, hence the dashed lines between the DAQ blocks. The DAQ blocks are also shown overlapping the 'LabVIEW (PC)' portion, as the PC physically communicates to the DAQ device (by USB cable in this instance), and the DAQ outputs are specified in LabVIEW.

A Laplace Transfer Function control system block diagram of a similar system is discussed and shown in Chapter 7. The LabVIEW program implementation is shown in Figure 6.24 on Page 85.

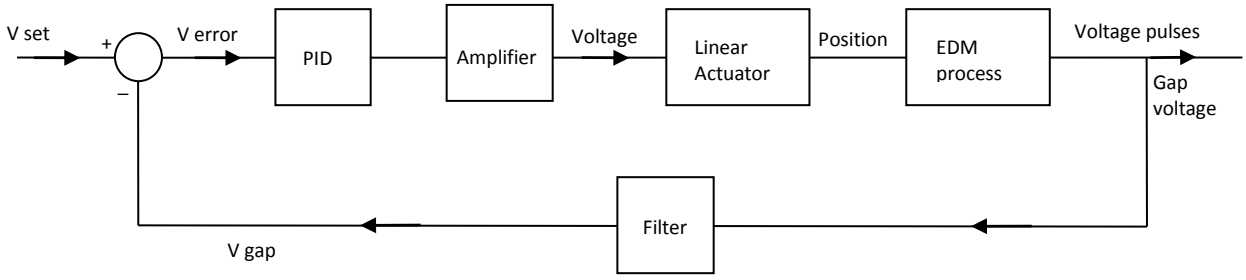


Figure 6.21 Simplified Block Diagram of Simple Control System

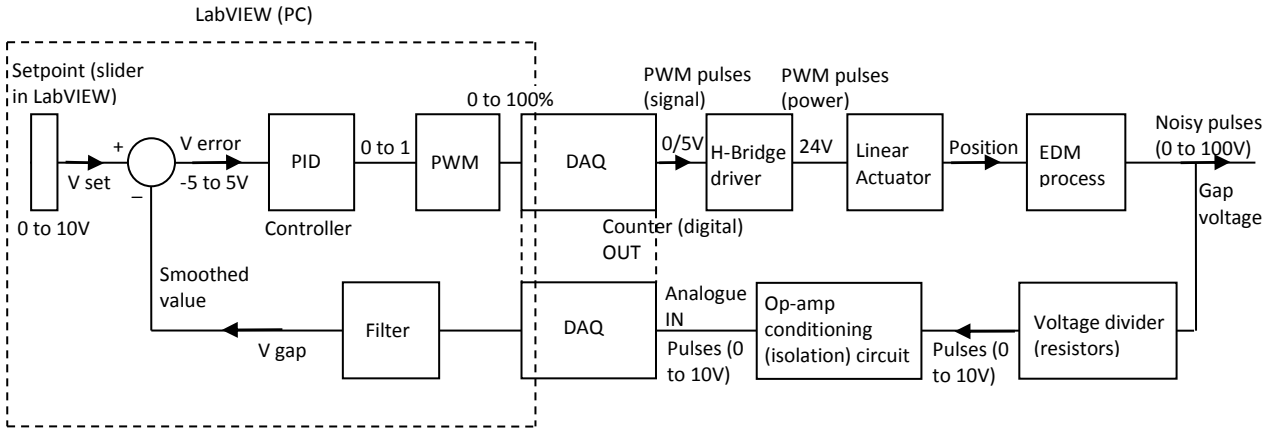


Figure 6.22 Block Diagram of Simple Control System showing Physical Equipment

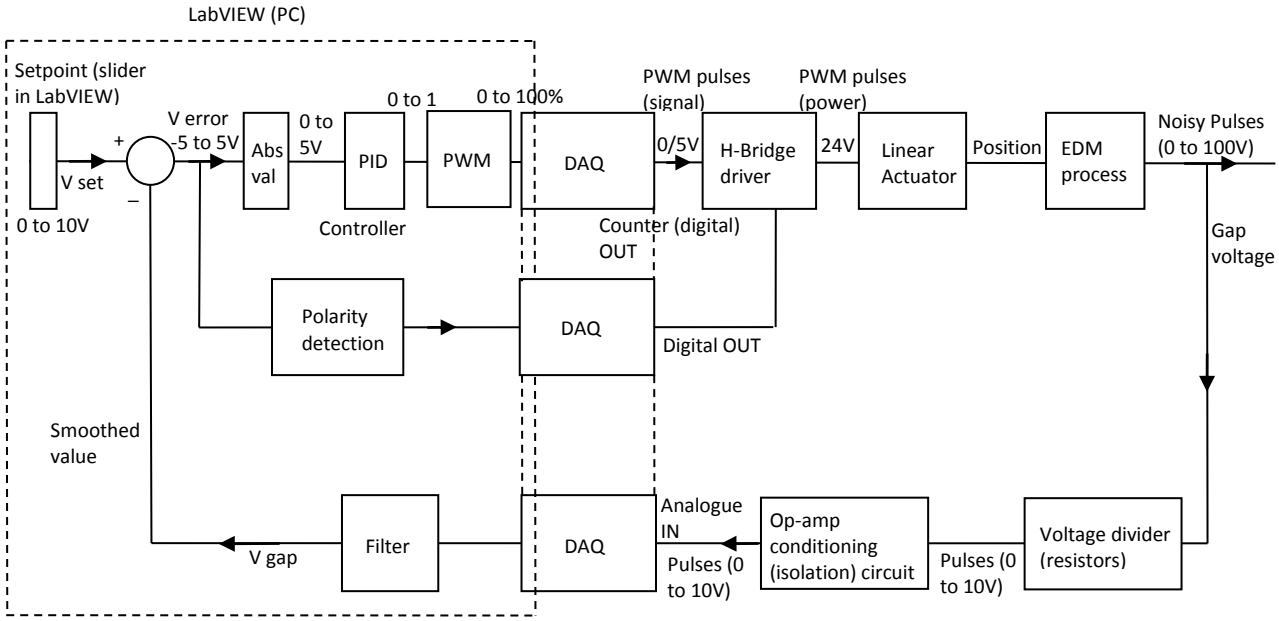


Figure 6.23 More Realistic Block Diagram of Simple Control System showing Physical Equipment

In the absence of friction, and if the EDM electrode is orientated for horizontal movement i.e. not downwards (which is not usually the case), then, since the voltage of the actuator produces a current which produces a force (torque in the case of a rotating motor) which produces an acceleration which produces a velocity which produces (over time) a distance (angle in the case of a rotating motor), and hence a correction of gap voltage, such a control loop, even without an integrating component in the controller, is essentially 'self-integrating', as opposed to a loop which specifies a position directly as output – a steady state error would persist if no integration aspect was included in the

controller (in LabVIEW) in such a case. Nevertheless, it should be mentioned that even so, such a loop (i.e. non-integrating) could still function satisfactorily, at least in terms of control stability; and even in terms of actual gap voltage/distance achieved, the voltage achieved can be physically seen by the operator (on Voltmeter on EDM machine's panel), who can then adjust the set point voltage until a satisfactorily output voltage results (the set-point adjustment being by virtue of a physical knob on the EDM machine's panel). This is of course important in terms of determining surface finish and material removal rate, etc. In this sense, the operator is the final feedback loop in the control system, taking action as need be according to the difference between the desired result and the actual result, minimizing the error.

If the electrode is arranged for vertical (plunge) movement, and/or if significant friction is present, then applying a voltage to a motor or actuator, proportional to gap condition error, will result in a steady-state error, i.e. a difference in true gap voltage (or distance) and the setpoint voltage (or distance). The vertical arrangement means that the weight of the moving part of the actuator, and the electrode holder and electrode, must be overcome by the actuator, just to maintain a specific position. I.e. a force must be generated by the actuator, which requires an armature voltage, to produce an armature current, to produce this force. Having no gap error implies having no controller output action, in the case of proportional control (with no control action offset). Thus, a non-zero steady-state error must exist, in order to generate this voltage to produce force, just to maintain position, since the armature voltage is simply the error multiplied by a gain. A simple way to minimize this steady-state error is to add an offset to the controller, i.e. controller output is proportional to the gap error, plus some fixed amount. This fixed amount can be chosen to produce the amount of voltage required to overcome the weight of actuator and electrode (but would ideally require adjustment if different electrodes were used). Steady-state error produced by friction, however, cannot really be compensated for by this method, since the friction action can be in two possible directions, depending on the current travel direction, or 'tendency to travel' direction, of the actuator. This effect is minimal in the case of the linear actuator, but in the case of a motor producing linear motion by means of a ball-screw, could be significant (worse still if a lead-screw is used instead of a ball-screw).

Proportional (P) Control

In the simplest version of the simple, single loop control system, proportional control only is used, as shown in Figures 6.24 (block diagram) and 6.25 (front panel) below.

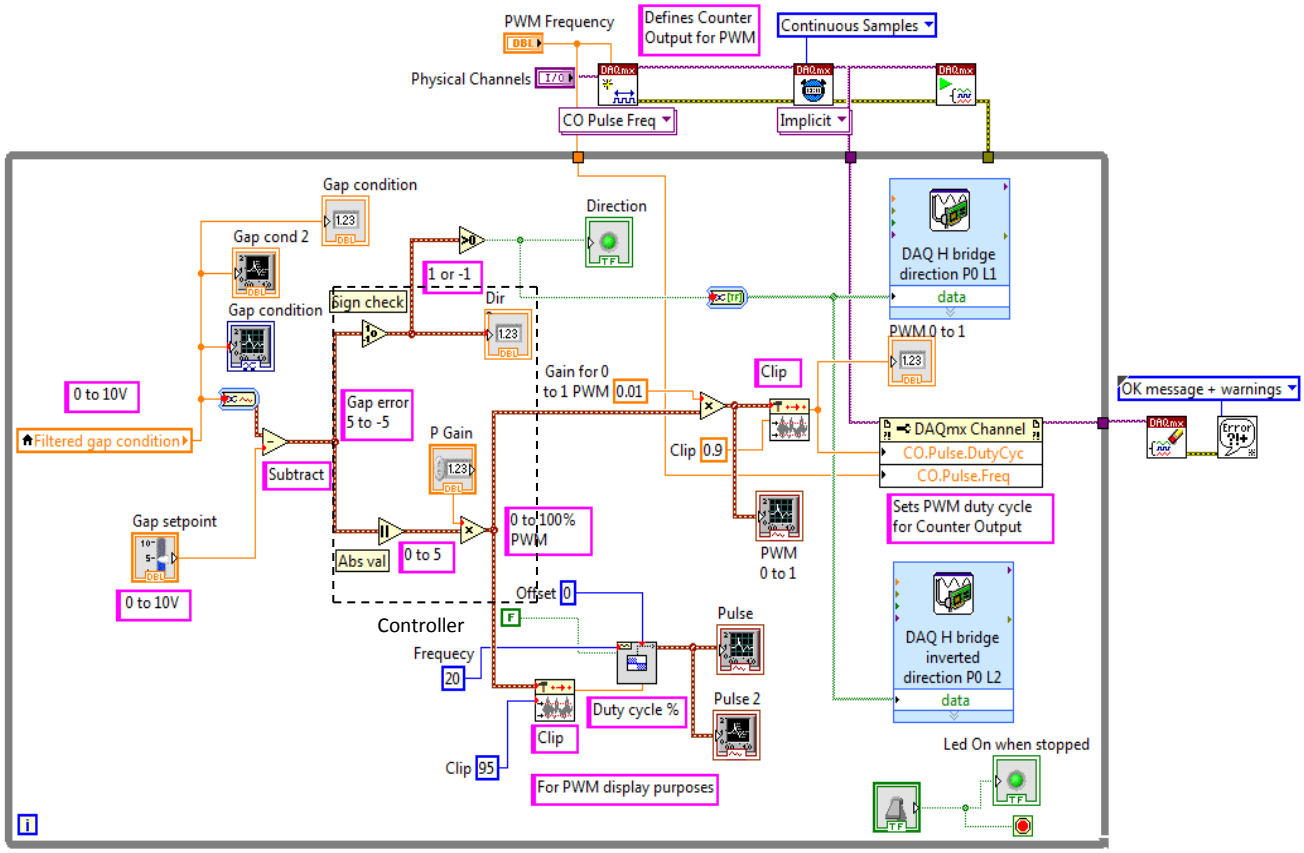


Figure 6.24 LabVIEW Block Diagram Program for Simple Control System, Proportional Control

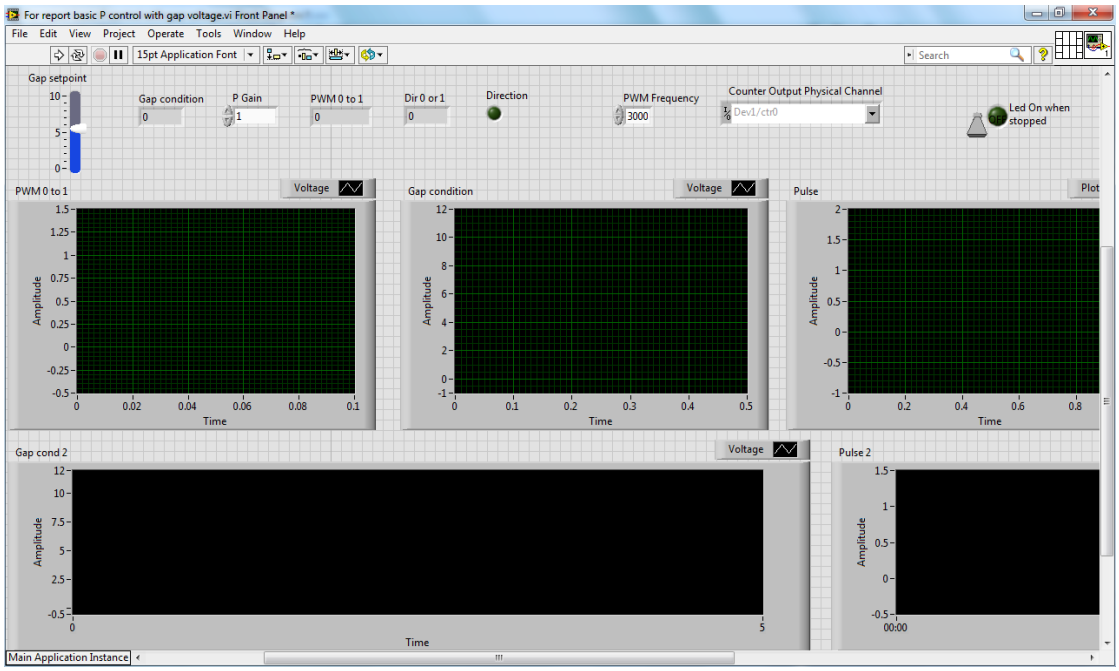


Figure 6.25 LabVIEW Program Front Panel for Simple Control System, Proportional Control

The program shows ‘Filtered gap condition’ as an input. This variable is a Local Variable getting its value from the filtering Do loop (refer to previous section).

The filtered gap signal is often referred to as a ‘voltage’, for simplicity. Naturally though it is just a numerical variable in LabVIEW having a 1 to 1 correspondence with the numerical ‘filtered gap signal’, where the input to the filter program loop is a variable (called ‘Gap voltage’) having a 1 to 1 correspondence with the true analogue 0 to 10V voltage signal being received by the DAQ device. I.e., no scaling of the input voltage signal was implemented where the input is defined for the port receiving the voltage signal.

This filtered gap voltage is compared to a setpoint voltage (specified on the front panel by means of a vertical pointer slider), i.e. a subtraction function is used. This difference is the error signal of the control system, and in this case is naturally a voltage variable. This error signal is multiplied by a gain K_P (P Gain on the diagram), and the result determines the PWM duty cycle % (or fraction). The PWM is represented, for display purposes, by a square wave function block outputting to a graph which can be seen on the front panel; the duty cycle of the square wave is specified by the ‘controller’ (in terms of the definition in the simplified block diagram of the control loop, shown above) output, where the controller is the portion shown in broken lines on the LabVIEW block diagram.

The function block above and to the right of the square wave generator is the block which determines the PWM duty cycle (as a fraction in this case), of the physical output port selected on the ELVIS DAQ board. This output port contains a physical counter which is used to create the PWM, without burdening the processor executing the LabVIEW program. A ‘Clip’ function is used to maintain the PWM duty cycle within the range slightly less than 1, as LabVIEW returns an error if the PWM variable is 1 or more. Thus if the error multiplied by the gain(s) produces a value higher than 1, the PWM is limited essentially to 1, and this can be seen as controller saturation, i.e. no more control action is taken, regardless of whether the gap error increases further or not. During testing, the upper clip value was often made significantly less than 1, thus limiting the voltage and acceleration of the servo device, for safety reasons (the safety of the equipment mainly, that is; unlike a conventional motor, the linear actuator shaft can ram into the ends of its travel and damage the device).

Testing with LVDT

For initial testing purposes, the LVDT position sensor (0 to 10V) was used as a surrogate for the gap voltage (with the actuator not mounted on the CNC machine, but fixed to a board for testing purposes, as shown in Figure 6.26 below).

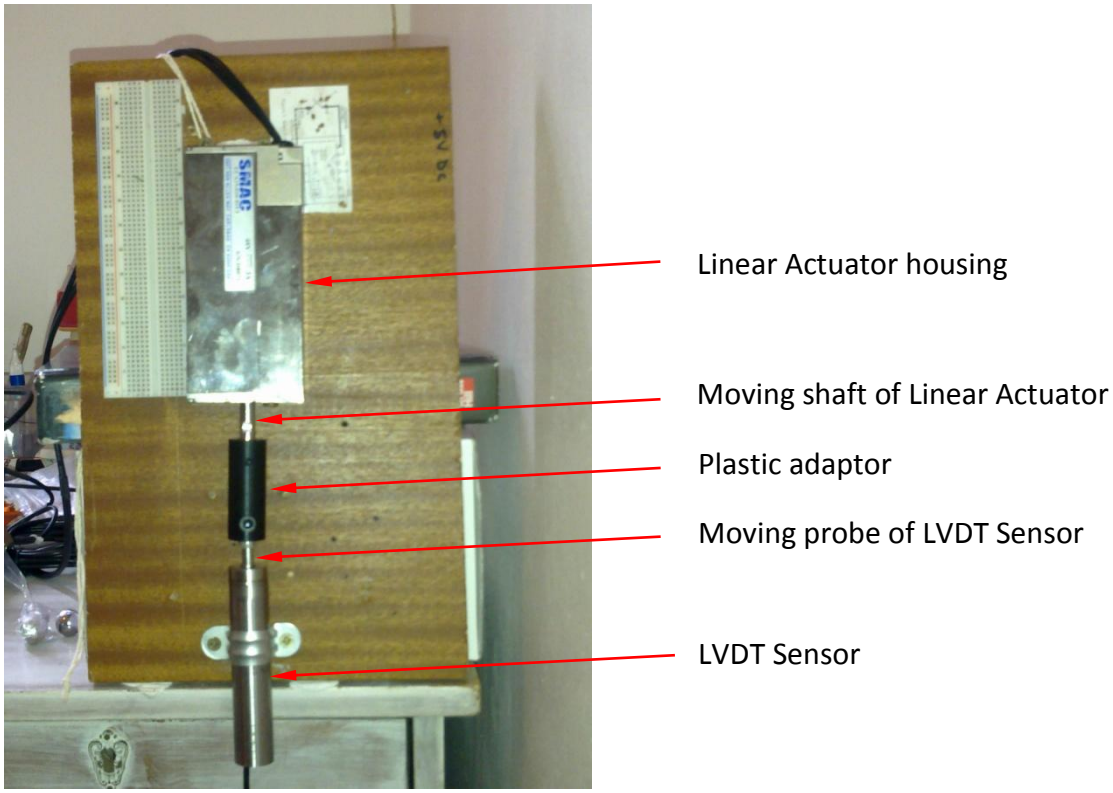


Figure 6.26 Linear Actuator connected to LVDT Shaft

The LVDT sensor was positioned opposite the linear actuator, its shaft attached to the shaft of the actuator, such that as the actuator extended, the voltage output from the LVDT would vary. Naturally the gain chosen would be a little different to the gain used when using the gap voltage, since 0 to 10V on the LVDT corresponds to a movement distance of 1mm, but a change of 10V from the EDM gap detection circuit does not necessarily correspond to 1mm of electrode travel, but is dependent on the method of detection used, and the particular circuit and signal processing used. Also, the polarity of the signal must be taken into account – as the actuator shaft is extended, the LVDT sensor shaft becomes retracted, which corresponds to an increase in output voltage, whereas when the actuator extends its shaft during EDM, this reduces the gap distance, which corresponds to a reduction in gap voltage.

The use of the LVDT position sensor as surrogate for gap condition signal naturally assumes a linear relation between movement and voltage change, which is not necessarily the case for real EDM. Nevertheless, it was felt likely to be a reasonable indicator such that performance of the control system could still be assessed using this method (the main purpose is to achieve gap condition stability; also, if it is stable at a point, the signal can be assumed linear over a short range of travel; the electrode would not necessarily always be at the same point though, in terms of distance from the workpiece, if a steady state control error exists and setpoint was not adjusted by the user to compensate). If a fixed setpoint is specified, then, when using the LVDT, the actuator will move to a fixed point within the 1mm range of the sensor and stop there, whereas on a true EDM system, firstly there is a lot of noise due to the nature of the process (EMI etc.), and there is the stochastic nature of the process, where debris is being built up and dissipated continuously, both of which tend to produce tiny advance and retract movements to follow the noisy signal. In addition, even with a fixed setpoint, the actuator gradually (on average) advances, as the workpiece becomes eroded. So with real EDM the control process never tends towards a stable, fixed, position, after a certain amount of initial oscillation, as is common with a positioning system. With the LVDT as surrogate, a high gain could cause a continuing oscillation which would be more representative of EDM, and to some degree the control system can be evaluated just with the LVDT signal; but to get a more realistic system i.e. to more realistically evaluate the performance of the control system, a noise signal should be purposefully added to either the setpoint, or to the feedback signal. This will produce a system with similar characteristics to real EDM control. Such a noise signal is shown in certain other LabVIEW block diagrams later.

Since a PWM output is desired from the DAQ, as input to the motor driver, the controller (in LabVIEW) output needs to be separated into magnitude and sign (polarity), since an H-Bridge requires a (digital) ‘direction’ input (i.e. dictating negative or positive voltage applied to the armature, corresponding to advance or retract), and a ‘PWM’ input (a train of digital pulses). Some H-Bridge devices accept an analogue (always positive) input representing PWM duty cycle (fractional On time of the signal vs. Total cycle time), and then perform the conversion to real PWM digital pulses to drive the H-Bridge switching elements On or Off. The H-Bridge driver used in this project however required a true PWM input, which was thus produced by the DAQ device. A suitable IC could have

been used to provide conversion from an analogue output from the DAQ, to PWM, but it was simpler to generate the PWM in LabVIEW i.e. as an output from the DAQ.

An error signal usually tends to swing positive or negative (if the input signal can potentially be higher or lower than the setpoint). After multiplication by the gain, the controller output would still have a negative to positive range. The amount of voltage to be applied to the actuator is dependent only on the magnitude of this signal (zero PWM means no voltage to the actuator, i.e. do nothing, and 100% PWM means maximum voltage to the actuator, i.e. largest response). But the actuator must be told whether to retract or to advance, in accordance with whether the EDM gap is too large or too small at a particular instant. The positivity or negativity of the error signal naturally describes this, and is thus used to determine the H-Bridge polarity, i.e. via the 'Direction' input. The magnitude eventually dictates the PWM duty cycle of the respective digital signal (square wave counter output) on the DAQ device, and the sign determines the state of the relevant output port connected to the H-Bridge direction pin input. Some H-bridges have only one direction input, such that motion is necessarily dictated (if PWM is non-zero), whereas others have two direction inputs, and one must be High while the other Low, for voltage to be applied to the armature coil (and other combinations of these inputs determine either 'free-wheeling' or 'braking' of the actuator/motor).

The particular H-Bridge used requires two inputs for 'direction'. For the requirements of the process, it was thought that voltage of one polarity or the other should always be applied to the actuator, i.e. no free-wheeling or braking is required, therefore two direction outputs were used on the DAQ device, the one being the digital inversion of the other, where this inversion is specified in the set-up parameters of the particular DAQ output channel. Inversion could of course be created by a function block in LabVIEW, but it was felt that there was less likelihood of there being a slight time delay between the changes in state of the two outputs, if inversion was specified in the channel set-up. This delay would have most likely been negligible, although in some instances (i.e. with some H-Bridge devices), higher switching losses might occur. Some H-Bridges require that the two direction inputs never be high simultaneously or low simultaneously, in which case the immediacy of state change is more important, but in this instance would not be a concern. In one of the final programs used in testing, an inverting Schmidt trigger IC was used to create the inverted signal, in which case only one DAQ direction output port

was used. A LED on the front panel LabVIEW screen indicates the state of the direction signal, i.e. the actuator polarity.

Proportional and Integral (PI) Control

Since for testing purposes the LVDT sensor was used as a surrogate for gap voltage, a steady-date error would exist, even if the actuator were in a horizontal position, as the LVDT employed a spring return as mentioned above (thus requiring a force to be produced by the actuator to counteract F = kx of the spring, when the sensor is displaced). The stiffness of this spring was quite significant (presumably the stiffness is made quite large by the manufacturer to ensure fast response when oscillatory measurements are being taken). Thus a controller offset was provided (added), as well as an integrating component, as shown in the Figure 6.27 below. The integrating component has its own gain (K_I; 'I gain' in diagram). These functions would be useful for real EDM, as the electrode was orientated vertically. Figure 6.28 overleaf depicts the front panel of the program.

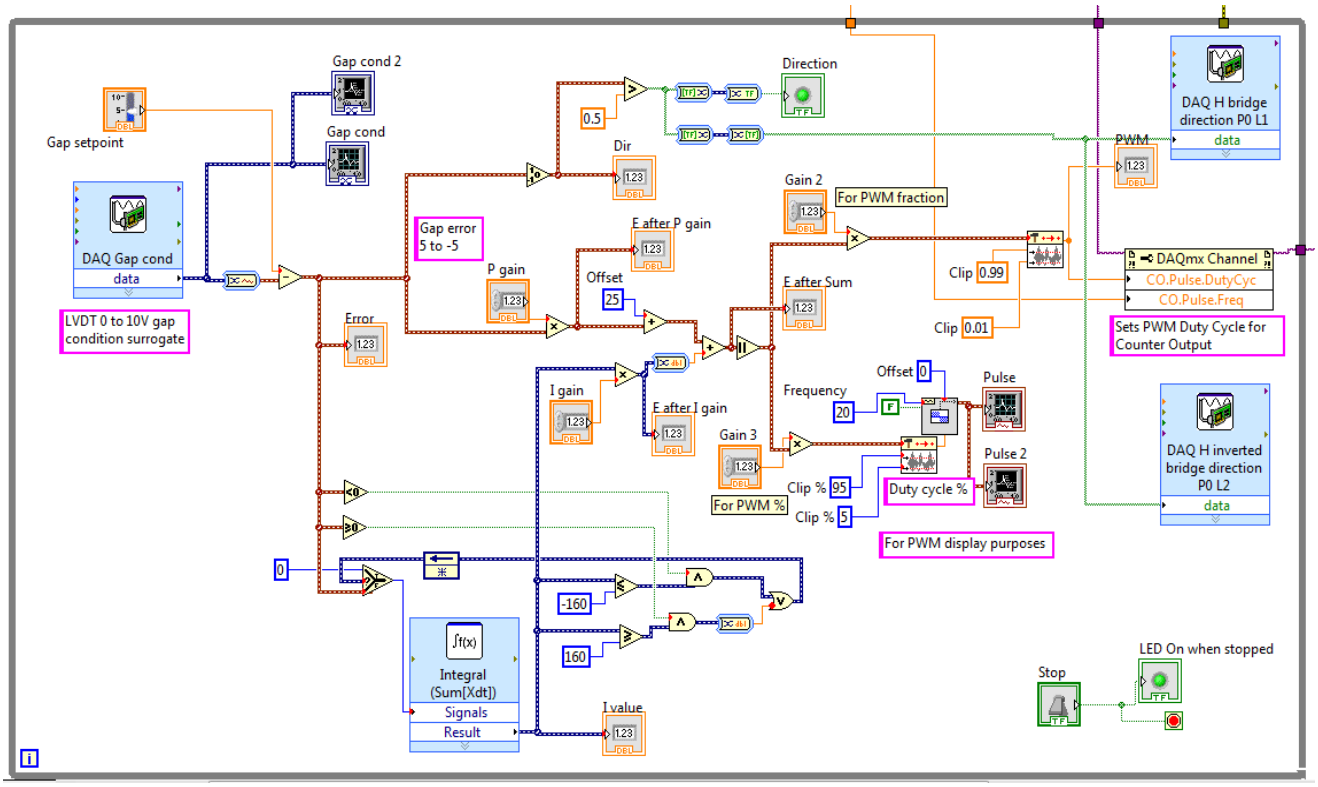


Figure 6.27 LabVIEW Block Diagram Program for Simple Control System, Proportional and Integral Control

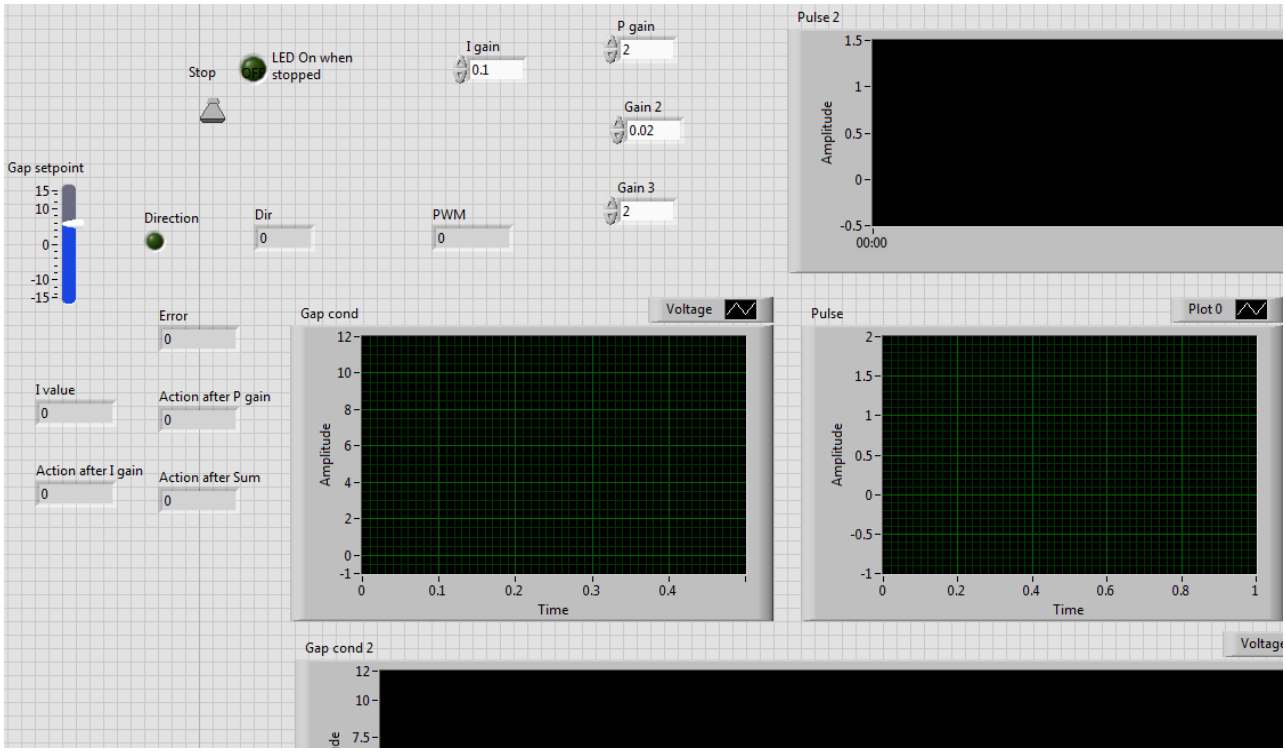


Figure 6.28 LabVIEW Program Front Panel for Simple Control System, Proportional and Integral Control

After each gain (P and I), a numerical indicator block for the value was used so that the individual contributions to the control action could be easily seen on the front panel. The combined control action (Sum of P, I, and Offset) is also shown on the front panel. Gains were made easily adjustable by proving input blocks on the front panel for each.

When using an integration (I) component in a controller, part of the control action is proportional to the integral of the error signal over time; thus the longer an error of a certain polarity exists, the larger the control action produced by this component. Usually integral control action is limited by the fact that, as over and undershooting of the controlled variable occurs, the error signal is constantly changing sign, thus the integral values builds up only a little before decreasing because of the change in sign of the error. In fact the integral action is forced to be within certain limits (assuming controller saturation is not occurring), as the larger the integral action is, the more the process tends to minimize the error (since this is the purpose of the control loop anyway).

'Integral wind-up' can occur when either a very large error exists for even a short period of time, or if an error of a given polarity exists for an extended period of time. This tends to happen under control saturation conditions, i.e. when the controller is trying to take

action to correct an (large) error, but cannot achieve enough actuator action to effect this correction. Alternatively, it can occur when a sudden, large, change in setpoint is applied to the system. Integral windup means that the value of the integral control action (contribution to total controller action), becomes extremely large, generally saturating the controller. The main problem caused by integral windup is that, even when the error returns to a small value, meaning the controlled variable is approximately the desired value, a large control action still exists (as not enough time has passed to bring the value of the integral component down to a small value), which would tend to drag the controlled variable away from its desired value, even when the cause of the integral windup is no longer present.

Limiting Integral Windup

To limit integral windup, the following strategy was devised. Essentially it involves making the input to the LabVIEW integration function block zero, once a certain desired maximum allowable value of the integration block output has been reached, such that the integration value does not continue to grow. However when the error signal changes sign such that the integration value would naturally start to decrease again, one does not want to prevent this tendency, i.e. one no longer wants zero as the input, and instead one again wants the error signal itself to be fed into the integrating block. This philosophy is significantly different to simply 'clipping' the value of the integration block output so that its effect on the control action is limited, as the real value of the integration block could still be extremely high, such that even when the error signal changes sign and starts reducing the integral value, a long time may need to pass before the integral value reaches the clip value, let alone a smaller, more normal, value. During this time, control is adversely affected as the process may be requiring action in a certain direction, but the controller may still be producing action in the opposite direction due to the large integral action component (which could be much larger than the proportional component, which would be 'trying' to produce action in the opposite direction).

As part of this philosophy, one needs to take into account the sign of the current integral value (output of the integration function block), as well as the sign of the current error signal (the usual input to the integration block). When the Integration value is positive, AND the error is positive, AND the specified (positive) maximum allowable integral value has been reached, then zero is to be fed into the integrator. Likewise when the

integration value is negative, AND the error signal is negative, AND the negative ‘maximum’ value has been reached, then zero is to be fed into the integrator. Under any other circumstances, the error signal itself is to be fed into the integrator as per usual.

To implement this philosophy to limit integral windup, one can see on the LabVIEW block diagram (Figure 6.27 above) the integration function block, and the adjacent logic functions and ‘feedback node’ (arrow pointing to left, with asterix below). One can also see the ‘decision’ function block, which chooses whether to fed the error signal into the integrator, or zero. The limits for the integration action can also be seen. One further development was that the error size itself, where being fed into the integrator, was limited by a clip function (not included on the particular block diagram shown), such that no excessively large value could be added to the integrator in any one iteration, which could potentially take the integrator value to a very high number, before the logic would prevent the integrator value from increasing any further, by feeding zero into the integrator. This was because in some instances, a very large error existed, albeit for a short time, causing the controller to saturate. The ‘clip’; function preventing a large error value from entering the integrating block is also not included in the particular block diagram shown.

It should be mentioned that, although the LVDT position sensor was used as a surrogate for gap voltage, one could just as easily have used the position of the actuator as determined by its own encoder, as a surrogate (this was done in some of the later programs developed/tested). However, the position as determined by the encoder is a digital number, whereas the LVDT output is an analogue value (as would be the output from the true voltage detection circuit), and was thus likely to have some small fluctuation even when the sensor was stationary; thus it was felt a more realistic substitute for gap signal, which would be noisy and never stable. Also, since the sensor produced a voltage in the range 0 to 10V, which was similar to what was expected from the gap voltage detection circuit, it seemed suitable to use the LVDT as opposed to the encoder position. That said, there is of course a difference introduced in the system when the LVDT sensor is used – it provides an opposing force proportional to displacement because of its spring return; this effect is not present in true EDM.

6.6.2 Implementing Velocity Control

Another control strategy considered and implemented (again suitable only for single-axis control) is to use the EDM gap voltage error (or a surrogate error), to create an instantaneous velocity command for the actuator (as opposed to the error simply producing, after PID implementation, an instantaneous armature voltage). This is a cascade system, in that the difference between the gap voltage and setpoint produces an error (voltage error), and this error is then used to command a desired velocity from the actuator, i.e. it is used to create an instantaneous setpoint for another control loop, which is a velocity control loop. Thus there is an outer loop using gap voltage feedback, and an inner loop using velocity feedback. This arrangement is shown in (simplified) control block diagram form in Figure 6.29 below.

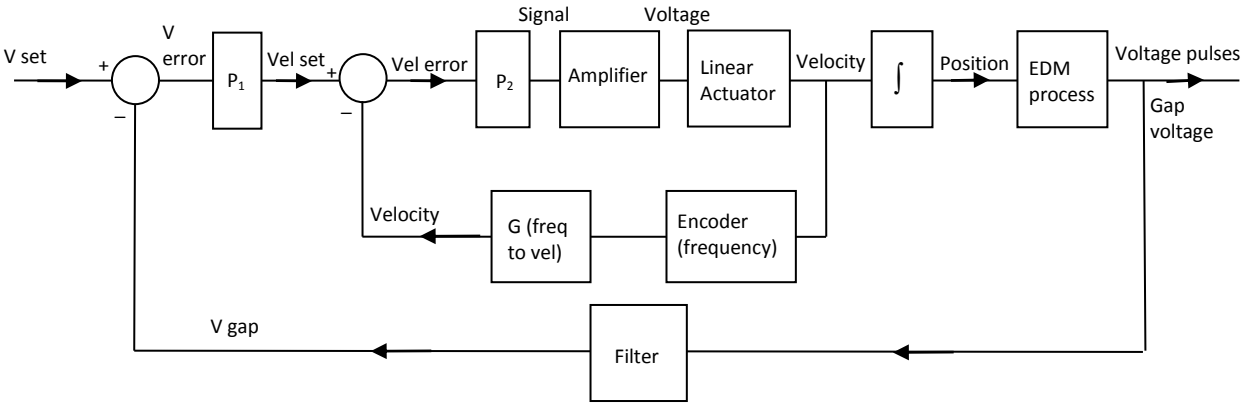


Figure 6.29 Simplified Block Diagram of Velocity Control System

The true velocity of the actuator, for use in the inner control loop, is determined by means of the actuator's encoder. The frequency of pulses from the encoder is a direct (proportional) measure of velocity.

Alternatively, if the encoder signals are being used to increment (or decrement) a counter to produce a Count variable, then, since this would be proportional to position (1 encoder pulse represents 1 micron distance in this instance), the derivative of the Count variable would be proportional to velocity. This method was also investigated and implemented (discussed later).

This method of control produces a more responsive system, as once a velocity (proportional to the gap voltage error) is commanded, the velocity feedback loop makes sure that this command is obeyed (naturally with some error); i.e. whatever armature

voltage is required to achieve the commanded velocity, is forced onto the armature by the velocity feedback loop (which compares the commanded velocity with the true instantaneous velocity, to create a velocity error, and generates a control action (armature voltage) proportional to that error). In the previous method outlined, a gap voltage error produces essentially a proportional armature voltage. A DC motor's speed is proportional to the armature voltage applied, but only under steady state conditions. Thus there will be a tendency towards a higher velocity when a higher armature voltage is applied, but it will not generally be proportional, as the motor (or linear actuator in this case) is never really in a steady state condition. So if one considers say a step change in gap error, with proportional control this would produce a step change in armature voltage. This in turn will produce a tendency to increase velocity, but it will certainly not be instantaneous (i.e. a step), due to the inertia of the motor/actuator – it will tend to be more like a ramp function, with tapering-off effect. With velocity loop control, however, a step change in gap voltage produces a step change in commanded velocity. This does not produce just a proportional armature voltage, but instead produces whatever armature voltage is required to force (as quickly as possible) the motor/actuator velocity to the commanded velocity. Thus if the gain of the velocity loop is fairly high, the velocity response of the motor/actuator will be more like a step (i.e. more immediate), as opposed to a ramp-like effect. That is, the voltage to the armature will be forced to spike as needed to achieve the commanded velocity, as any difference between commanded velocity and true velocity will be detected by the velocity control loop and amplified by the high gain, forcing the desired motor response.

To visualize this method of control in an EDM system, one again needs to bear in mind that the system is of a noisy, oscillatory nature. I.e. at one instant the gap distance is too large, say; this produces a velocity in the direction to reduce this gap; overshoot invariably occurs, and the gap becomes too small; this produces a velocity in the direction to make the gap larger; overshoot occurs, and the gap is now too large; and so the process continues, bearing in mind that there is, in addition, EMI noise and the stochastic nature of the process. One might have thought this method would be inherently more unstable than the previous method described (since actually in EDM a position is desired, not a velocity – which generally can be considered to cause overshoot). But presumably due to the more responsive nature of the control loop, this was found not to be the case; if anything, it was more stable than the previous method described. Granted, the stability does also depend on the gain values chosen. One of

the final control programs developed also has a velocity loop element, thus this aspect is discussed again later (section 6.7.2).

In order to use the quadrature encoder pulses to determine velocity, the frequency of the pulses is needed. However, the direction of movement also needs to be determined, for use in the control loop. An option is to decode the encoder signals in LabVIEW, and determine the direction in LabVIEW. However, it was felt that it would relieve the PC processor to some degree if the encoder signals were decoded externally (to Pulse and Direction signals), and then read into the DAQ device (in two input ports). For this purpose an HCTL-2021 decoder IC was utilized. Control block diagrams of the system are shown in Figures 6.29 and 6.30 below. The first shows that the encoder direction (and the H-bridge direction) is required; the second shows in more detail the interrelation between the encoder, decoder, DAQ device, and LabVIEW itself, in the feedback path of the inner (i.e. velocity) loop.

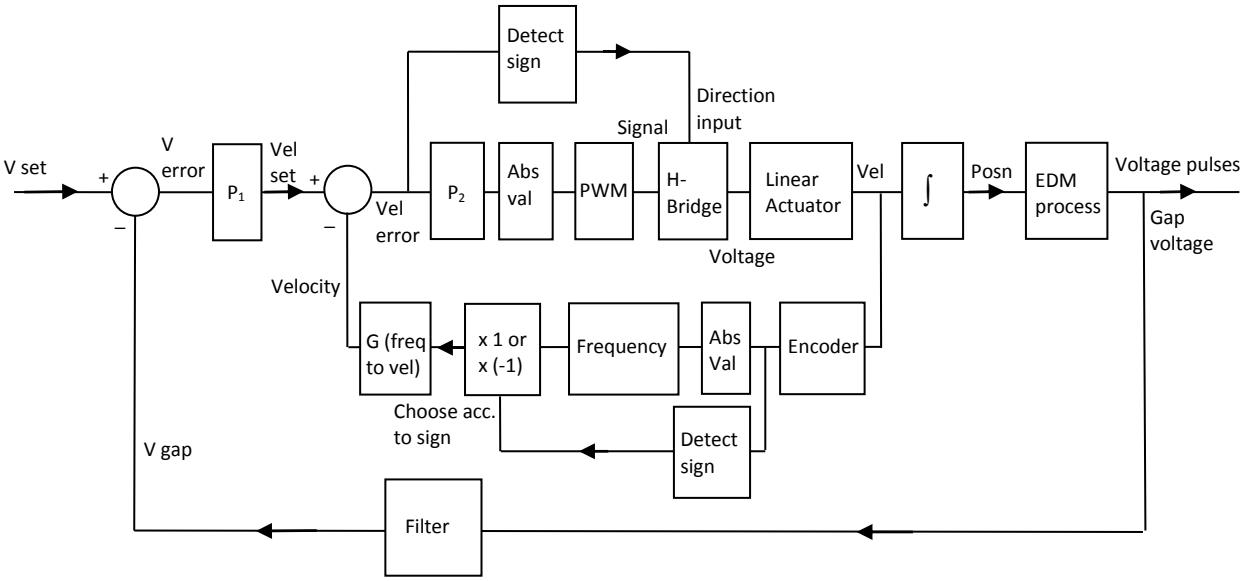


Figure 6.30 Block Diagram of Velocity Control System

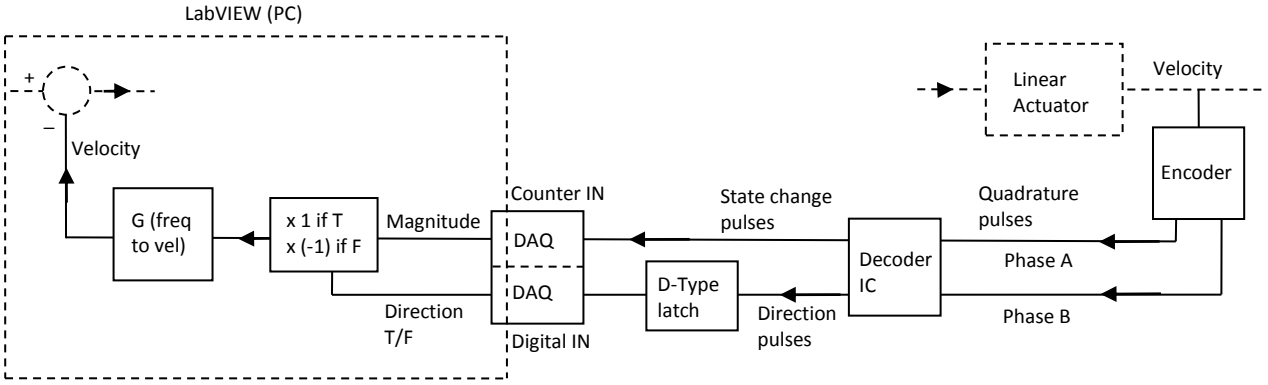


Figure 6.31 Block Diagram of Inner Loop Feedback Path of Velocity Control System, showing Physical Equipment

A schematic circuit diagram of the decoder and D-Type flip-flop latch arrangement is shown in Figure 6.32 below (the purpose of the lower D-Type flip-flop is explained later).

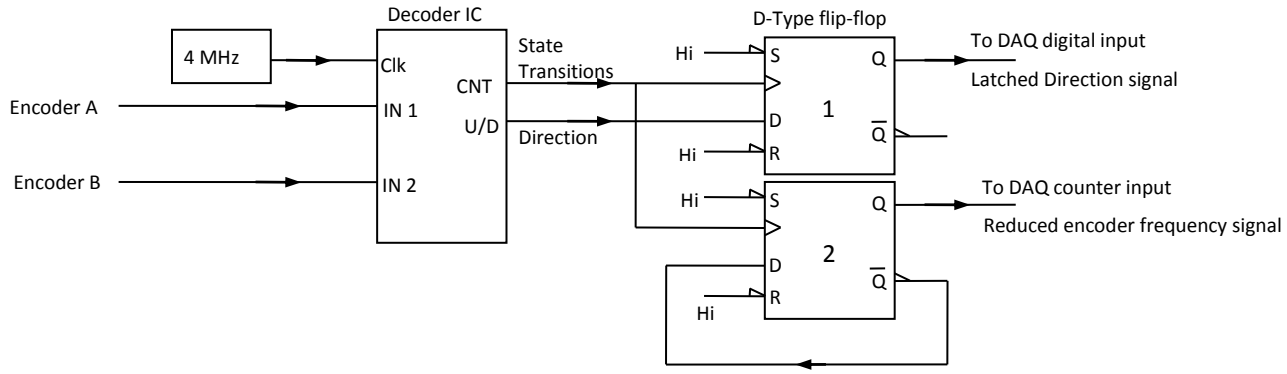


Figure 6.32 Schematic of Encoder Frequency and Direction Latch Circuit

The decoder IC requires a high frequency Clock input, which was provided by a 4 MHz resonator. The resonator frequency needed to be at least an order of magnitude higher than the frequency of pulses expected from the encoder. The 'Pulse' (CNT) output of the IC gives a pulse on each state transition of the encoder (i.e. on each of the encoder phases, A and B). The state of the Direction output is held high or low to indicate the encoder movement direction, during the time of the state change pulse on the other output (the Direction output state being present slightly before, during, and slightly after, the state change pulse on the other output). However, between state change pulses, the Direction output state always reverts to Low; thus if its state is read into the DAQ device during this time, erroneous data will be received. Thus the direction state needed to be latched during this time, either in LabVIEW or in hardware. It was decided to use a D-Type flip-flop in hardware for the latching function. In a D-Type flip-flop, the data present on the Data input becomes latched into the IC when the Clock input receives a rising edge. Thus the Direction output from the decoder was connected to the Data input, and the Pulse output was connected to the Clock input. In this way, the encoder direction state was held during the time between encoder state change pulses, and could be read continuously into the DAQ device for use in the LabVIEW program.

The picture in Figure 6.33 overleaf shows the decoder and D-Type flip-flop latch circuit (on loose prototyping board), positioned on the ELVIS DAQ device. Also shown is the H-Bridge motor (actuator) driver. The actuator's encoder signals enter the loose prototyping board at the bottom right.

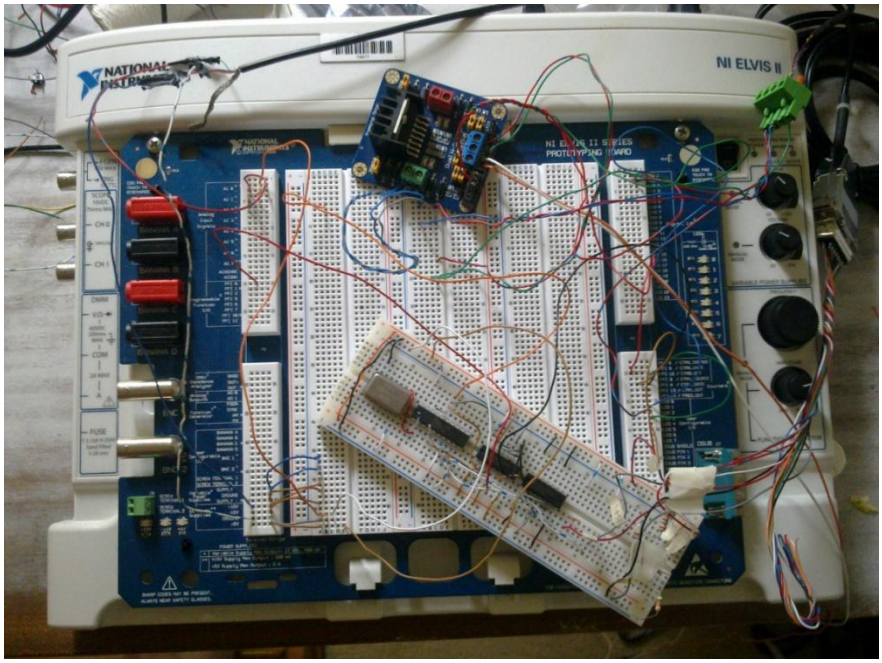


Figure 6.33 ELVIS DAQ Device, Decoder and Latch Circuit, and H-Bridge Driver

The LabVIEW block diagram program for the implementation of the velocity control system is shown in Figure 6.34 below, and the LabVIEW front panel follows in Figure 6.35 overleaf.

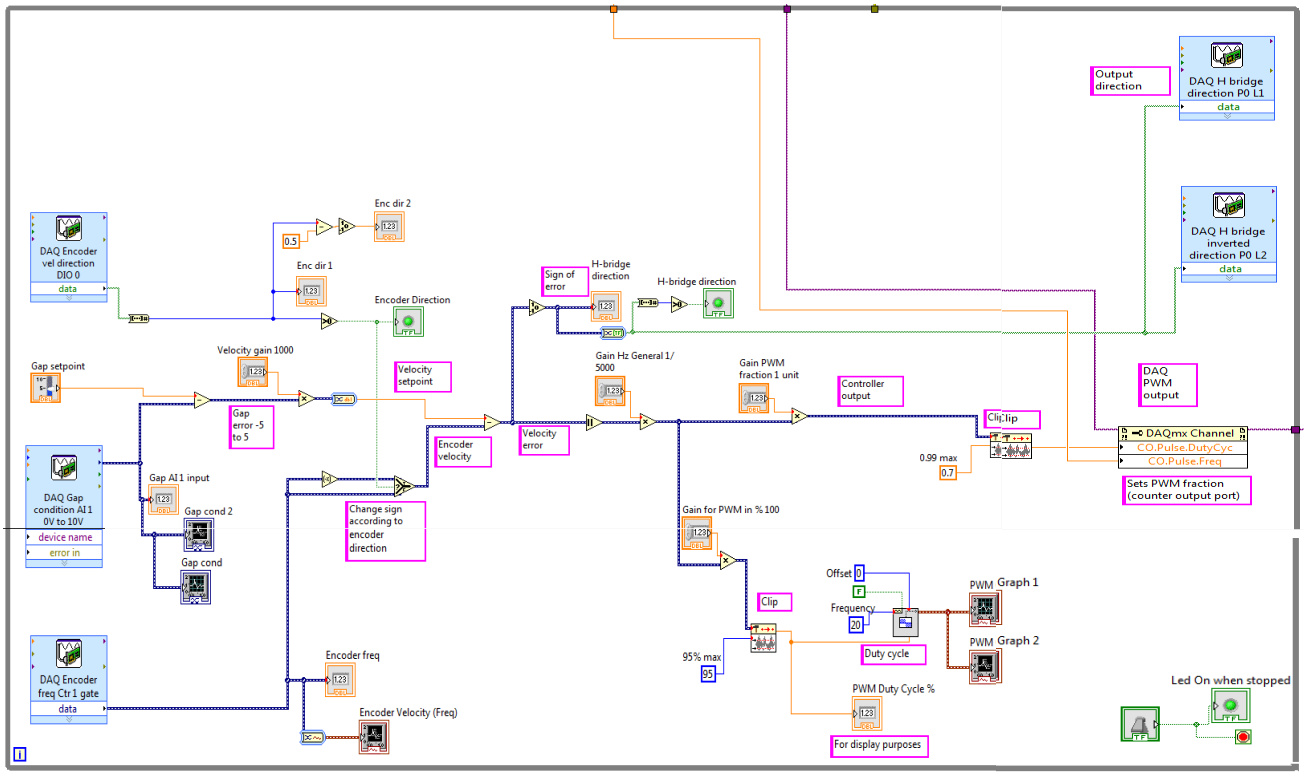


Figure 6.34 LabVIEW Block Diagram Program for Velocity Control System

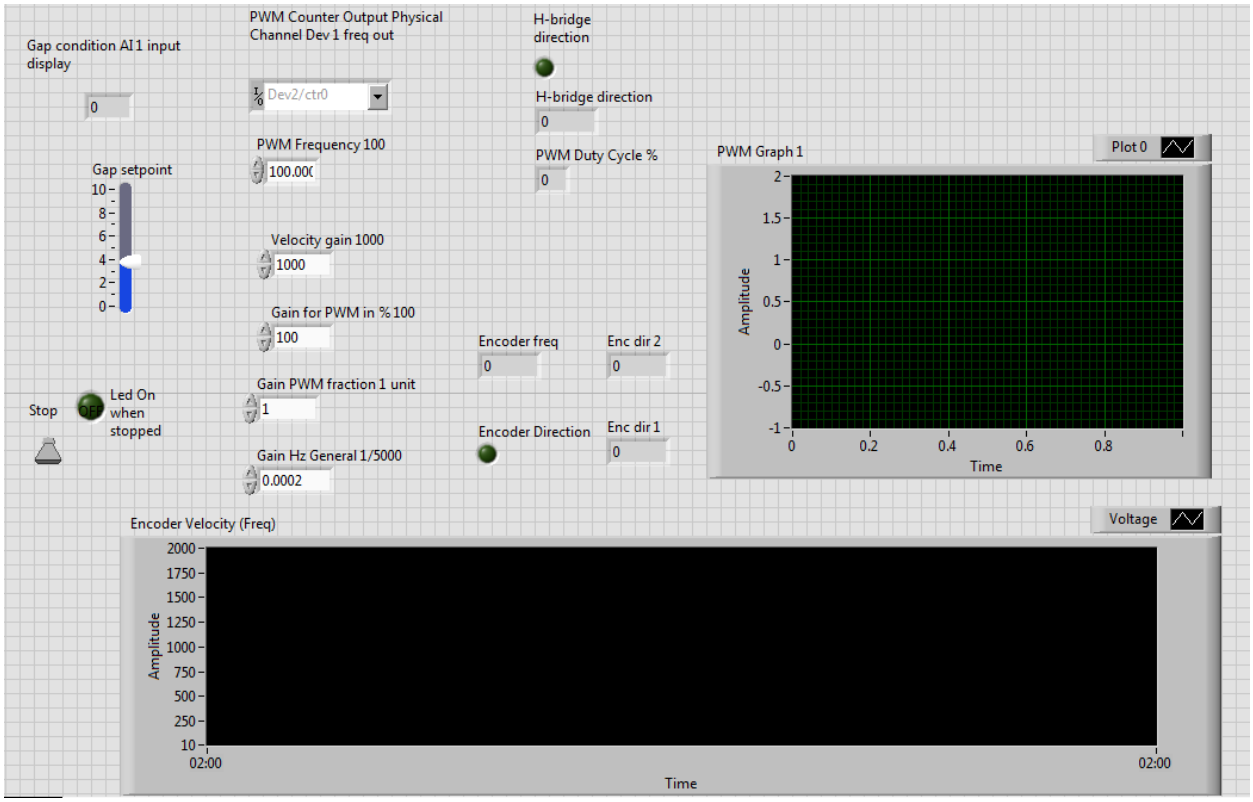


Figure 6.35 LabVIEW Program Front Panel for Velocity Control System

In the LabVIEW block diagram one can see that, depending on the state of the input measuring the encoder direction (as determined by the decoder IC), the frequency is multiplied by 1 or -1. This is important as the 'state change' output of the decoder only gives the magnitude of frequency (proportional to velocity), but the direction of the velocity is equally important. It can be seen on the diagram that there is no specific gain block to convert from frequency to velocity; this is because the encoder resolution happened to be 1 micron, so one pulse is 1 distance unit; also there is a gain block further downstream, where adjustments can be made.

For frequency measurement, a DAQ input port was configured accordingly. Frequency measurement is one of the selections available under the acquisition of digital signals options. A 1-counter method was employed; the counter essentially measures the time between consecutive pulses, in this case between consecutive rising edges of the input signal from the encoder. This time is of course the period of the signal, and its inverse obtains the frequency.

The encoder frequency measurement function was tested (initially) using a rotary encoder as substitute for the linear actuator's encoder, as the actuator was not available

at the time. Results of the testing were not entirely satisfactory; at higher speeds, the frequency measurement graph appeared to jump and not be smooth. A concern was that if there were slight imperfections in the spacing of the encoder divisions, then, although on average a good measure of frequency would be obtained, on consecutive pulses there may be large discrepancies (since the counter input port was measuring time between each pulse). Also the decoder did not provide the Pulse outputs at the exact time of the true state change occurrence – it provided output pulses only on an edge of its Clock input. Thus, if the clock frequency was not much higher than the state change frequency, slight errors would occur, as in some instances the IC would have to wait fractionally longer before providing a Pulse output, relative to the actual state change of the encoder, than in others (since the encoder state changes were a-synchronous to the decoder input clock). Also if the frequency sampling method in LabVIEW was not optimal, frequency 'jumps' would be apparent.

One way to make the frequency measurement smoother would be to create a 'divide by two' function in hardware, upstream of the DAQ counter input. This would not only create an average, which would already make the measurement more smooth, but would also reduce the frequency such that measurement at the DAQ input would be 'easier', and likely more accurate, in terms of sampling rates required.

The divide by two function was also implemented on a D-Type flip-flop (the same chip on which the direction latch was implemented was used, as it was a dual D-Type IC, i.e. there were two (independent) flip-flops on the same chip). The circuit diagram of the function is shown in Figure 6.36 below (the lower flip-flop has relevance for this function).

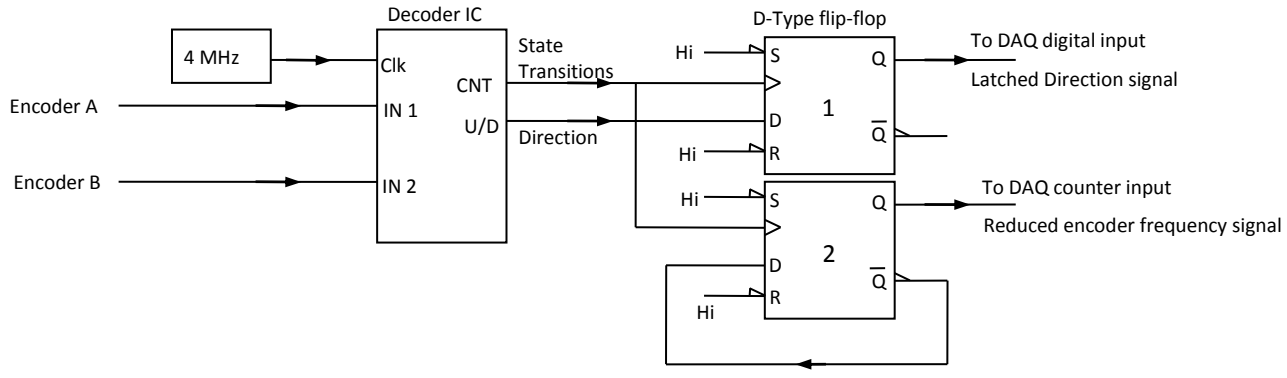


Figure 6.36 Frequency Divide by Two Function (D-Type Flip-Flop)

For this implementation, the inverse of the output of the D-Type latch (Q bar) is connected to the Data input of the same latch, and the flip flop is clocked (rising edge) with the state transition pulses themselves. The Q output was then fed to the DAQ device counter input, for frequency measurement (Q bar could also have been used). In this way, on the rising edge of one state change pulse, the output of the latch goes let's say High; then on the next rising edge it goes Low (because the inverse of the present state is 'waiting' at the Data input, ready to be clocked in with the state change rising edge). At the next state change rising edge, a High state gets clocked into the latch, again because the inverse of the present sate is waiting at the Data input. Thus every time the state change pulse goes Low to High, and Low to High again, the latch output goes High and then Low. And so a divide by two function has been created.

However, it became apparent later that the reason for the 'jumps' in the graphs displaying the measured frequency was more likely due to the sampling and update rates of the graphs themselves, than any of the potential reasons mentioned above. The output pulses from the decoder appeared evenly spaced on the oscilloscope used to check the decoder performance, and at low frequencies the frequency graphs were quite smooth.

One concern however when using the frequency measurement in LabVIEW, to determine velocity, is that since frequency was determined by measuring time between individual pulses, then once a specific pulses is received, the frequency value is only updated again when the next pulse is received. Thus if a 'next pulse' is never received, because perhaps the encoder has come to a complete stop, then the frequency value remains potentially high if the time between the previous two pulses was short. In other words, if the encoder comes to a complete and sudden stop, the frequency value never becomes zero, which it should if the true state of motion was being reflected, but may remain quite high. Also DAQ device 'time-out' problems occurred when the encoder became stationary, as the counter input is waiting for a 'next pulse' which never comes.

However, in a real EDM process, it was thought that these concerns might not be too serious, as actuator (hence encoder) motion is seldom, if ever, likely to stop completely, due the fluctuating nature of the gap conditions, and the noise on the gap voltage signal. Also, if actuator velocity was decreasing relatively smoothly during direction changes

(which would be happening all the time), then there would not be instances when the frequency value was erroneously held at a high value (which would of course tend to make the actuator shoot thorough the direction changes too ‘violently’, as velocity commands should ideally approach zero near the direction changes as they are proportional to gap voltage error which should be near zero at the direction changes – except that gap error jumps could occur anyway due to the nature of the EDM process itself). Nevertheless it remains possible that e.g. friction may prevent the actuator from moving when gap conditions are relatively stable, i.e. even though small velocity changes are being specified by the control system, thus this method of velocity determination was deemed not ideal, and it was found during testing (at least in preliminary testing, i.e. using the LVDT, with the actuator not actually mounted on the CNC machine), that the actuator did sometimes get ‘stuck’ and time-outs did occur.

Since some immediacy of measurement of velocity (frequency) is lost when using the ‘divide by two’ function (and discrimination/resolution is reduced), naturally it would be better to omit this function, if the frequency was actually being captured smoothly anyway. Also, there had been a concern that there would be a state changes ‘missed’ during encoder direction reversals, when using this method. This concern was later proved warranted, as when testing was performed using the linear actuator and its encoder, when the actuator was moved to the end of its travel and back again, each time it returned to its original position, the encoder Position value (discussed later) was found to be 4 units less than it was the previous time. This ‘missing’ of pulses would not be a concern for single axis control, where just velocity measurement is required, but it would be for multi-axis control, where the instantaneous position of each axis (actuator) is required. Since the direction reversals happen at a very high rate when using the true EDM gap voltage (even when filtered), this proved a serious problem for multi-axis control.

Thus later the D-Type latch divide by two function was omitted, and in addition, another method of obtaining velocity was used, namely where encoder Phase A and B decoding was performed in LabVIEW, producing a ‘Position’ variable, which when differentiated gave the velocity. (The encoder decoding program did not give velocity directly, only the accumulative encoder ‘Position’ value, which naturally takes forward/reverse direction (Up/Down counting) into account too; differentiating this Position value thus gave the encoder velocity, and its sign, not just a magnitude; thus using this method, it was not

necessary to capture a Direction signal from the encoder (or rather its decoder), as was the case when decoding in hardware.)

A control system block diagram for velocity measurement using this technique is shown below (just the velocity feedback i.e. inner loop is shown, as the remainder is the same as in Figure 6.30 on page 96). It can be seen that the decoding occurs in LabVIEW itself, and that a Position variable is produced, which immediately gets differentiated for use as Velocity.

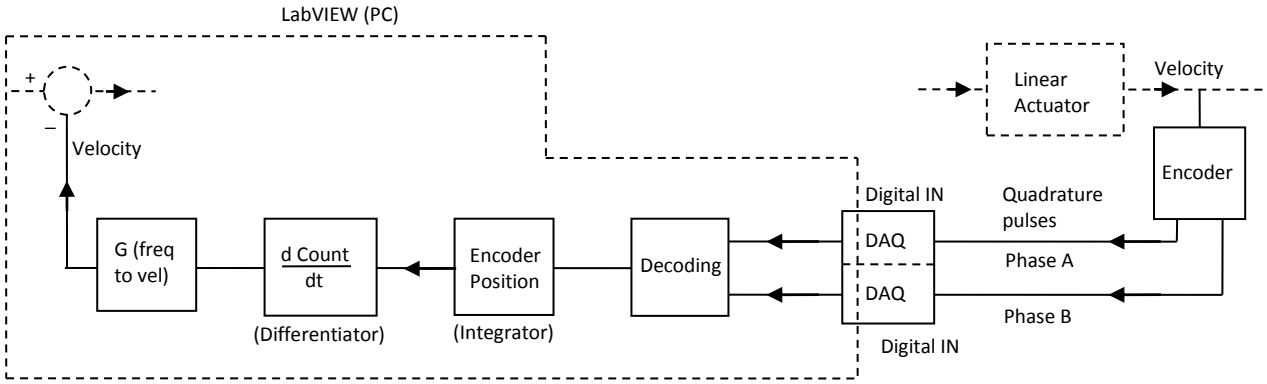


Figure 6.37 Block Diagram of Inner Loop Feedback Path of Velocity Control System, showing Decoding and Velocity Determination

Since the Encoder Position program (a standard LabVIEW VI available, for use as a sort of sub-VI in this case) runs in its own Do loop, the Position variable had to be extracted using a Local Variable, for use with the differentiating function, when this technique was implemented in LabVIEW. 'Position' and encoder 'Count' are essentially the same thing, just a multiplier has been applied to convert Counts (steps) to distance. Since the Sub-VI is actually for an angular encoder, the 'Pulses per Revolution' variable had to be specified such that the distance per step worked out correctly at 1 micron.

The LabVIEW implementation is shown in Figure 6.38 overleaf. Again, the sub-VI (Encoder Position) block diagram code (Figure 6.39) was extracted from the standard program and placed on the main program block diagram, for (presumably) faster execution.

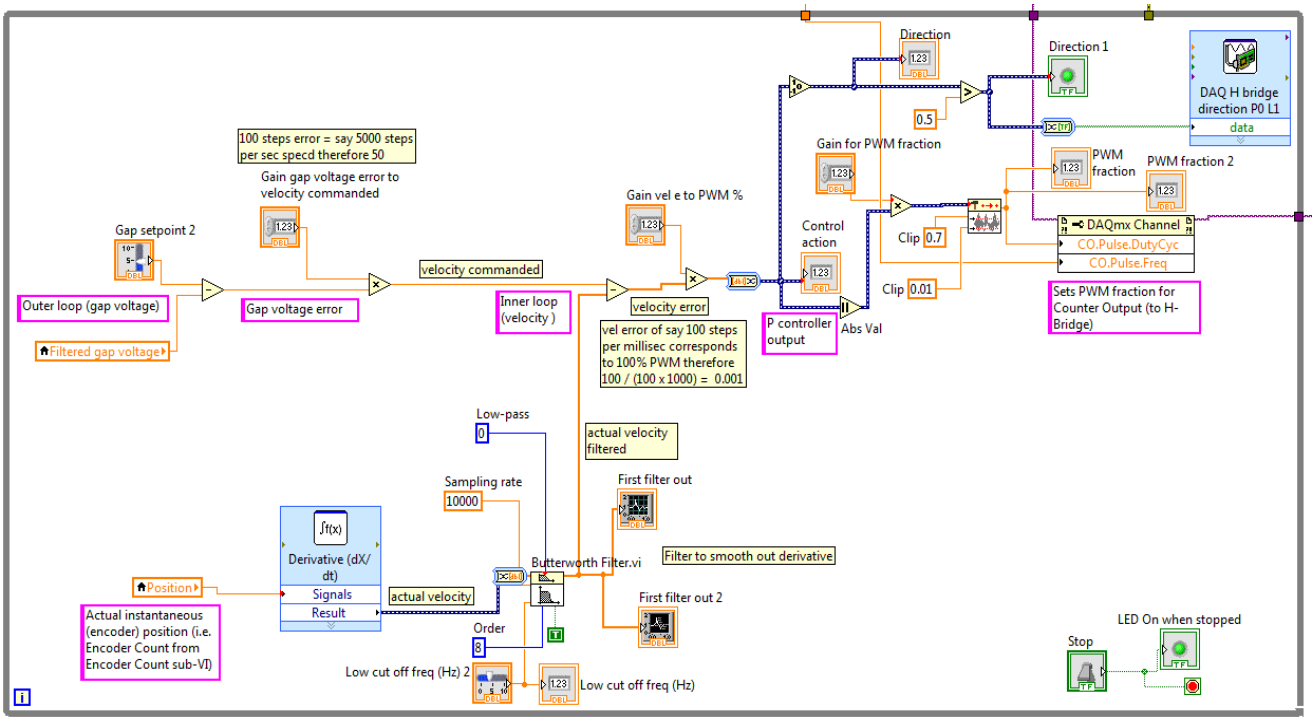


Figure 6.38 LabVIEW Block Diagram Program for Velocity Control

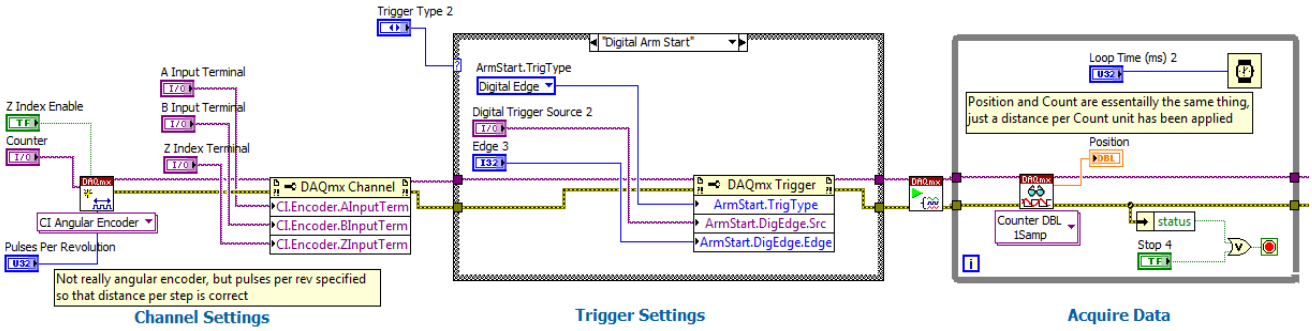


Figure 6.39 LabVIEW Sub-VI for Encoder Position

[Note, initially a Count Edges (sub) VI was used to provide the encoder position, as discussed in section 6.7.1 later in this Chapter This performed essentially the same function as the Encoder Position VI, only the output of the VI was Count, not Position. When using the Count Edges VI, the decoder chip and Direction (only) D-Type latch were maintained; the VI established whether to count up or whether to count down, depending on the state of a digital input port reading in the Direction signal from the D-Type latch; see section 6.7.1 for further discussion of this implementation.]

As mentioned earlier, this method of control as described is suitable only for single-axis control. This is because although (different) velocities can be commanded for each of

say two axes, proportional to the desired linear travel ratio of the axes, this will not necessarily produce the same ratio in terms of distance travelled, as the velocity commands would not be perfectly obeyed by the actuators due to different axis inertia values and different friction amounts, etc. (or any other imperfections including differences in gains of the control loops). I.e. there would be an accumulative error when trying to force a distance ratio by using a velocity ratio (velocity integrated over time does indeed produce a distance, but if the integration of each axis is slightly different, then there will be a distance ratio error produced).

Since multi-axis EDM is ideally desired as part of the project outcomes, attention was then turned to methods for ensuring that multi-axis EDM could be achieved, i.e. methods that take into account the ratio of movement of each of two (or more) axes, and these are discussed in the following section.

6.7 Multi-Axis Control Programs

For multi-axis EDM, it is naturally necessary not just to control the gap distance/voltage, but also to co-ordinate the movements of the axes. Two-axis EDM control will be considered as demonstrative of multi-axis EDM (the methods used can easily be extended to three axes). If a part being EDM'ed is to have a surface at an angle i.e. not flat or vertical, then a linear movement needs to be specified for this. I.e., the ratio of the displacements of the two axes needs to be the same at all times during this linear EDM process, and the process must stop when the specified end point of travel is reached. Since the process is made up of very small advance and retract movements, the ratio of these movements also needs to be the same as the ratio of the total distance of EDM desired, for the linear move. Thus linear interpolation of a sort needs to be employed, where the endpoints of the move are specified for each axis, and the intermediate points are determined by the interpolation method. In a normal interpolation process there is a time variable (or a variable that is proportional to time) that increments at a certain rate during the process until the end point of the linear move has been reached. In the case of EDM, however, the rate at which the process steps through the 'time' variable is not really crucial (the EDM process to a greater or lesser degree determines the overall and instantaneous velocity of movement), but the ratios of axis movements is crucial. The axes will be called Z and Y for discussion purposes (Z being a natural choice because EDM is usually has a downwards component).

Thus, in an EDM process, one can think of the ‘time’ variable being incremented and decremented during the process, as governed by the process itself; i.e. the process steps forward with the ‘time’ variable (incrementing), or retracts along the same path, while the ‘time’ variable is decremented. So in the case of EDM as here envisaged there there is not a direct time variable (or variable proportional to time), which always advances with time, but a ‘time-like’ variable that can increase or decrease as time progresses, needs to be created. Thus some control loop needs to specify tiny advance and retract commands, for each axis, to satisfy the gap requirements on a very short time-scale, and these commands can be seen as increments or decrements of the ‘time’ variable. As the process progresses, there will be on average more increments than decrements, as the workpiece material is gradually eroded away, and thus, on a much longer time scale, a specified value of the ‘time’ variable will eventually be reached (the end point of the linear travel specified). But on a short time scale, the process will be seen to be rapidly incrementing and decrementing the variable, as over-and undershooting occurs, and due to the fluctuating nature of the EDM gap (as well as noise).

Thus the process can be seen as many ‘mini’ position commands (for each axis, of pre-pre-determined ratio, and some ‘forward’ and some ‘backward’, always in quick succession). So there must be a position control loop (or rather a position control loop for each axis), executed within a gap voltage control loop (gap voltage dictates progress through the position commands, where the position commands are in the correct ratio to ensure that the desired linear path is traversed).

A typical block diagram of the multi-axis EDM control system developed to achieve this is shown in Figure 6.40 overleaf.

The multi-axis EDM LabVIEW programs themselves are designed to handle both axes simultaneously; however, as a second linear actuator was not available, the LabVIEW program shows where certain portions of the block diagram would be repeated for the second axis, but does not actually carry out any control of a second axis.

In the multi-axis EDM LabVIEW programs developed there is, for each axis, an inner loop which is the position loop, where an actuator position command (instantaneous

setpoint) is compared to the true position of that axis (by virtue of the encoder on that axis), and a position error is generated. In some instances the inner loop is not just a position loop, but a combination of position and velocity.

This position error then dictates the actuator armature voltage (PWM), in much the same way as gap voltage error did in the previous strategies described, by virtue of some form of PID controller (not necessarily with all three components), such that corrective action is taken to minimize the position error.

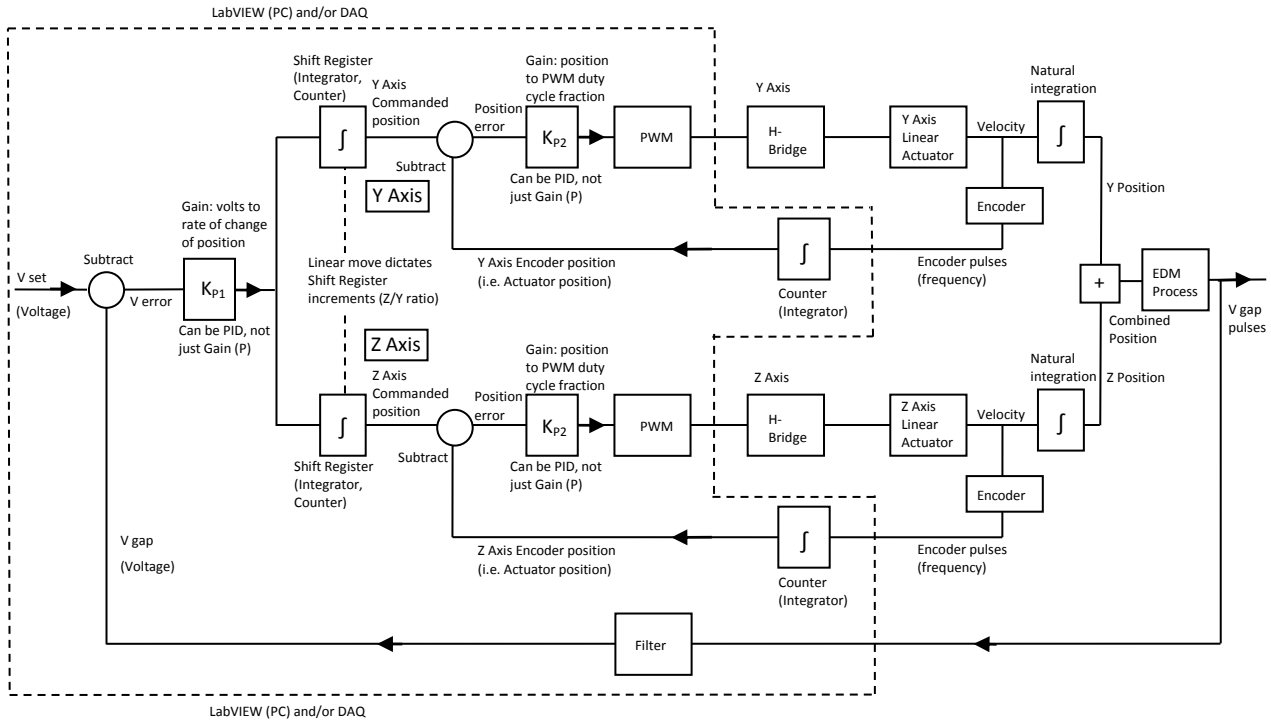


Figure 6.40 Typical Control System Arrangement for Multi-Axis EDM

To increment or decrement a ‘time’ like interpolation variable through its range till it reaches its specified final value according to the total travel requirements, a shift register (actually two shift registers, one for the Z axis and one for the Y) seemed the natural choice. Each shift register is arranged to either increment or decrement its variable, (which is the Z or Y axis position command to be fed to the position loop), according to whether the gap error itself is positive or negative (which is an indication of whether the electrode is too close or too far from the workpiece).

6.7.1 Multi-Axis Control: Shift Register Increment Rate Fixed

In the first version implemented depicted in Figures 6.41 – 6.45 below, instead of using a single time like variable to step through the process, it was instead decided to adjust the relative period of iteration of each of the two shift registers according to the ratio of 'steps' to be taken by the Z axis and 'steps' to be taken by the Y axis, during the same time period (i.e. so that both axes finish their movement at the same time, i.e. so that the correct end point of the travel is reached). This would achieve essentially the same objective. A Wait function (or Wait Until Next Millisecond Multiple) was used in each shift register to determine the period of iteration of each. The Wait period in turn is dictated by a consideration of the linear travel to be undertaken.

The user specifies a Z and Y distance to be travelled (i.e. specifies an endpoint of the movement). A fixed, pre-determined velocity (for Z and Y axes movement combined) for each 'step' is then used to calculate the periods of each shift register iteration (note there is no exact 'step' as such, for the two axes combined, as the two axes movements are a-synchronous in this instance, since their periods differ). Once a velocity of the axes, combined, is decided upon, the total time for the travel (if it were continuously in the same direction, i.e. not continuously changing due to gap conditions), is determined (time = distance / velocity). This time is then divided, for each axis, by the number of steps (encoder steps) to be taken by that axis (again, if steps were taken continuously in the same direction), to determine the period for each step (which becomes the Wait period for that axis' shift register). In this way the shift register values are incremented and decremented in the correct ratio's, and their values become the instantaneous position setpoints for the position control loops (and are passed on to the position loops by means of a Local Variable defined in each shift register). Thus if the gap condition error is positive, the electrode is advanced at a fixed velocity; and if negative, it is retracted at the same velocity. The electrode advances and retracts in quick succession, according to over-and undershooting of the gap distance, noise, and debris production. It is important to note that although fixed velocities are used, the EDM process still governs the progress of the electrode along the linear path, since the polarity of the gap error will be continually changing.

The LabVIEW program combines a number of modules. First a standard LabVIEW (sub) VI 'Count Edges' is shown in Figure 6.41 overleaf.

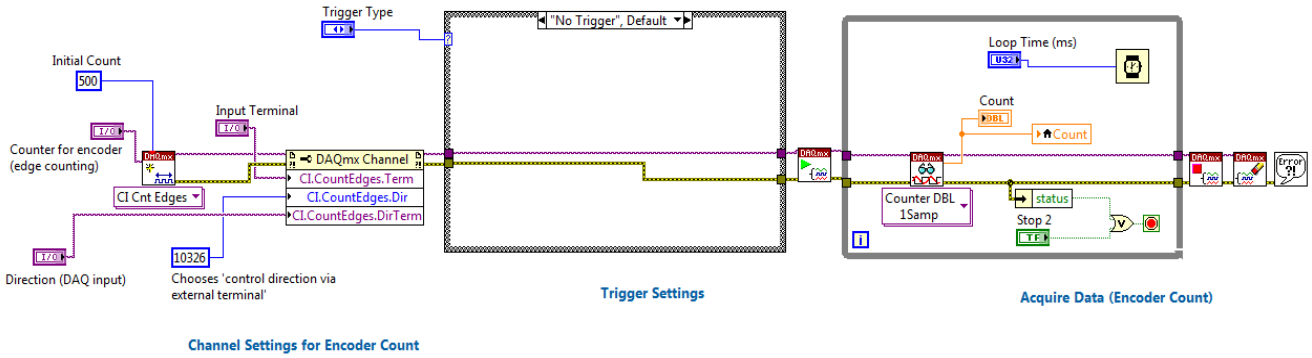


Figure 6.41 Count Edges sub-VI for Encoder Position Determination for Control Program for Multi-Axis EDM

This VI receives the pulses from the decoder IC (or in some instances the D-Type latch, when a 'divide by two' function was being employed), via the ELVIS DAQ device; the decoder in turn receives its inputs from the quadrature encoder phases A and B, of the linear actuator. The sole purpose of the VI is to keep track of the encoder position, by counting the pulses received. The '10326' number (input to CI.CountEdges.Dir, lower left) selects an option where an external port (digital input) dictates the direction of counting, i.e. Up or Down. This port receives its input from the D-Type 'Direction' latch (which is latching the decoder IC's Direction output). The Initial Count of 500 is chosen in this instance, to keep the Count value away from near zero, since the counter does not allow for below zero numbers. The Count value (Orange, near right) is passed to the position loop (inner loop) by means of a Local Variable.

The block diagram of Figure 6.42 overleaf shows the implementation of the outer control loop of the cascade control system, which monitors the gap condition (the gap signal filtering loop is not shown).

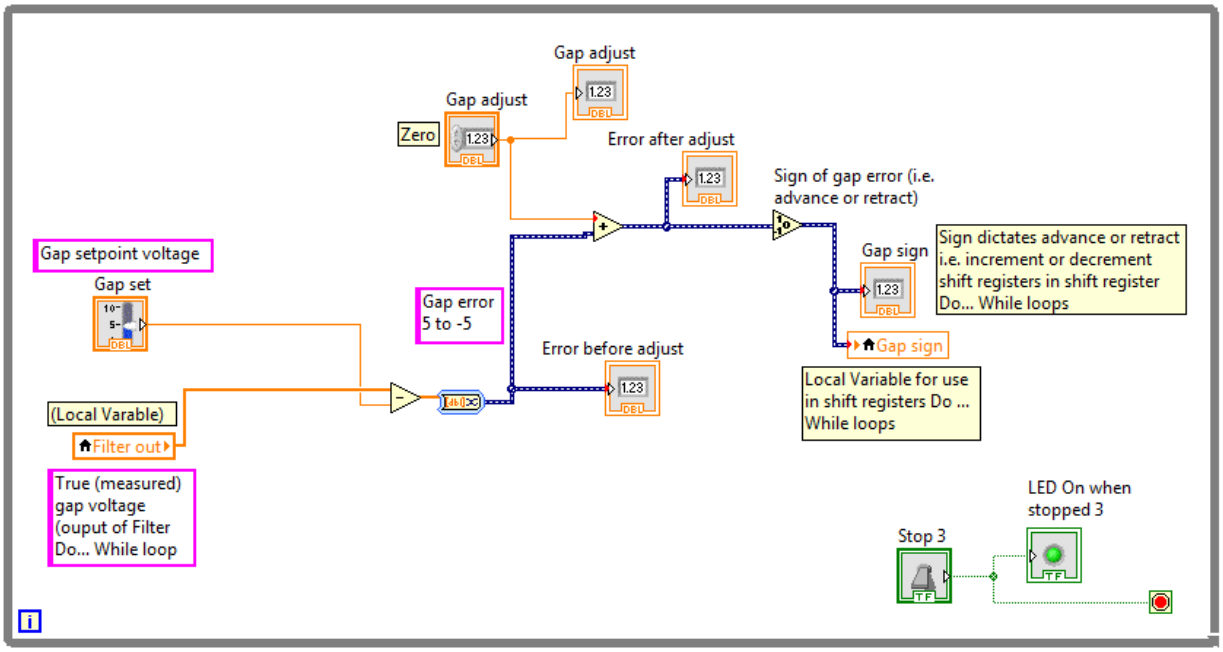


Figure 6.42 Gap Error Sign Determination Loop (Outer Control Loop) for Control Program for Multi-Axis EDM

The gap error sign determined in the afore-mentioned loop is then used in the shift register loops (one for each axis), to increment or decrement the register (i.e. adjust the commanded position setpoint for the inner position loop up or down). The Z axis shift register loop is shown in Figure 6.43 overleaf, first with its controlling inputs. (A similar loop is present in the program for the Y axis, but is not shown – the Z and Y Final Values specified are common to both loops). Then the shift register portion (only) is shown, zoomed in for clarity (Figure 6.44), and thereafter the controlling inputs portion (only) is shown (Figure 6.45, on the page following). The 'controlling inputs' in this context are the Z Axis Register Initial Value (500, to match the Count Edges initial value), the desired Final Value of the Z Register (specified by the user), the Z Axis 'Required direction check', and, most importantly, the Wait period for each iteration of the shift register Do…While loop.

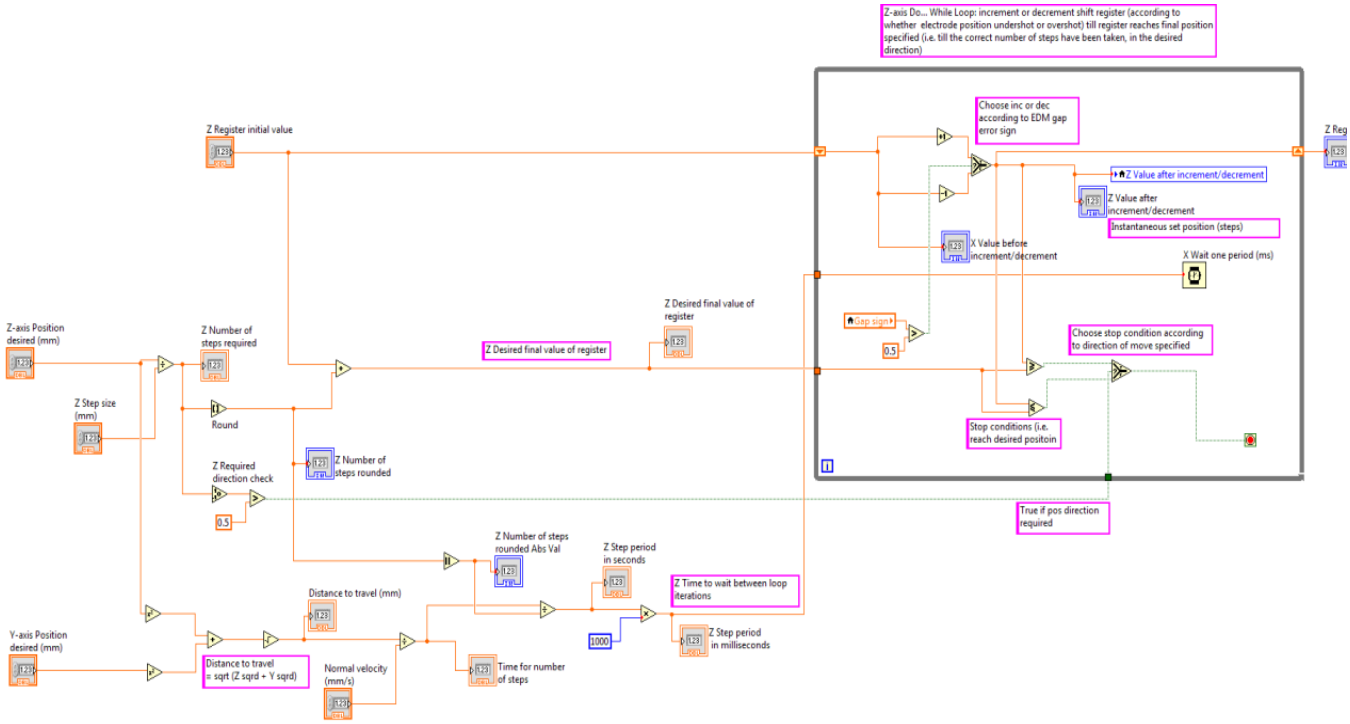


Figure 6.43 Z Axis Shift Register (and its Controlling Inputs) Portion of Control Program for Multi-Axis EDM

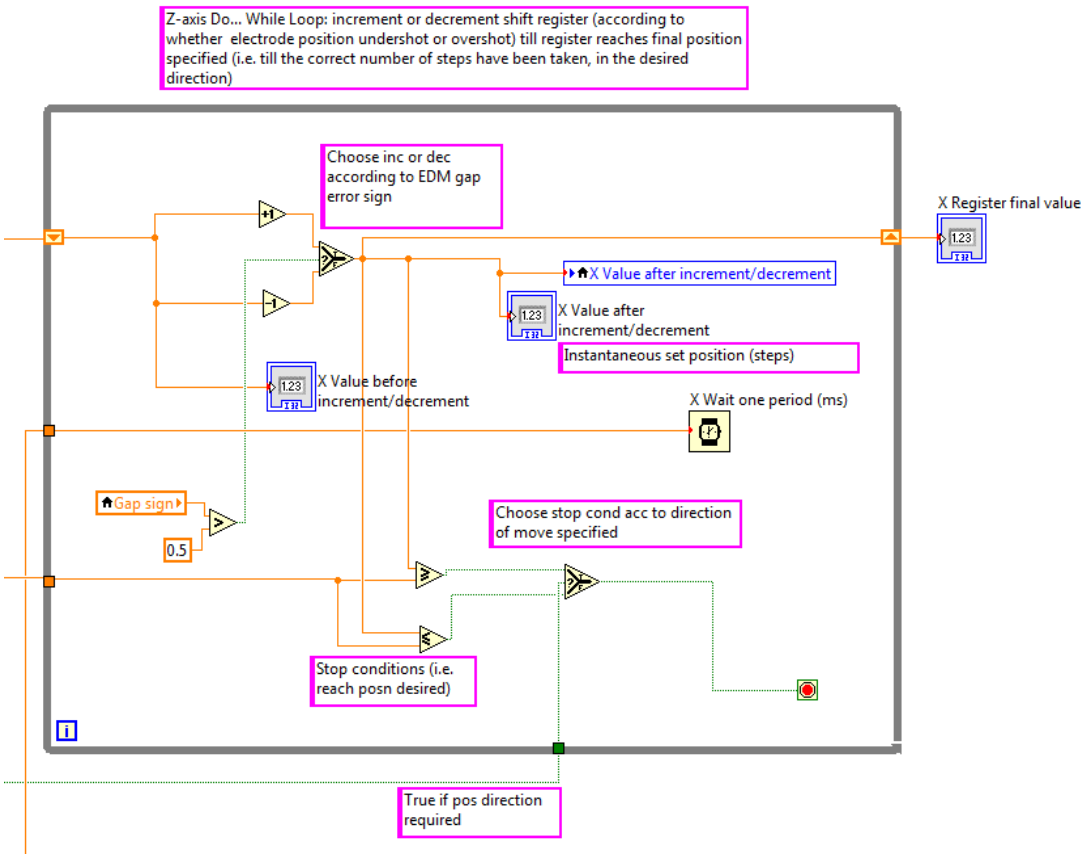


Figure 6.44 Z Axis Shift Register Portion of Control Program for Multi-Axis EDM (Zoomed)

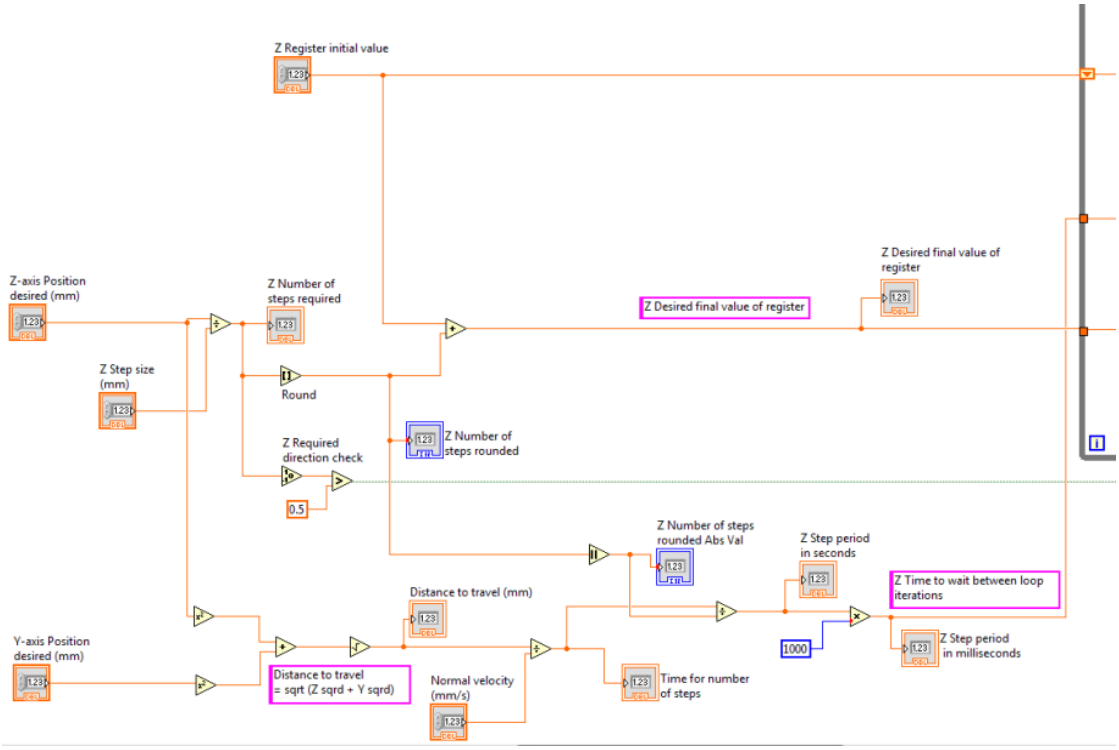


Figure 6.45 Z Axis Shift Register Controlling Inputs Portion of Control Program for Multi-Axis EDM (Zoomed)

The instantaneous value of the Z axis register is passed on (via a Local Variable) to the position loop (inner loop of the cascade system), where it is compared with the true position of the actuator (by virtue of the encoder), and control action (voltage to the actuator armature) is generated. The position loop portion of the program is shown in Figure 6.46 overleaf. In this instance it includes P, I, and D control action.

There are at least two problems with this implementation. Firstly, although the direction of the specified velocity of the electrode changes in accordance with the gap error sign, the specified velocity (i.e. the combined velocity of the actuators, which by Pythagoras is the square root of the sum of the squares of the individual actuator velocities) remains the same at all times, regardless of the size of the gap error, whereas it would be preferable for the specified velocity to drop, even approaching zero, as gap error approaches zero (to minimize overshoot).

Secondly, the smallest time resolution available for the Wait function in LabVIEW, (or Wait Until Next Millisecond Multiple) function, is 1 millisecond. Thus if the specified velocity dictates shift register increment periods on the order of only a few milliseconds,

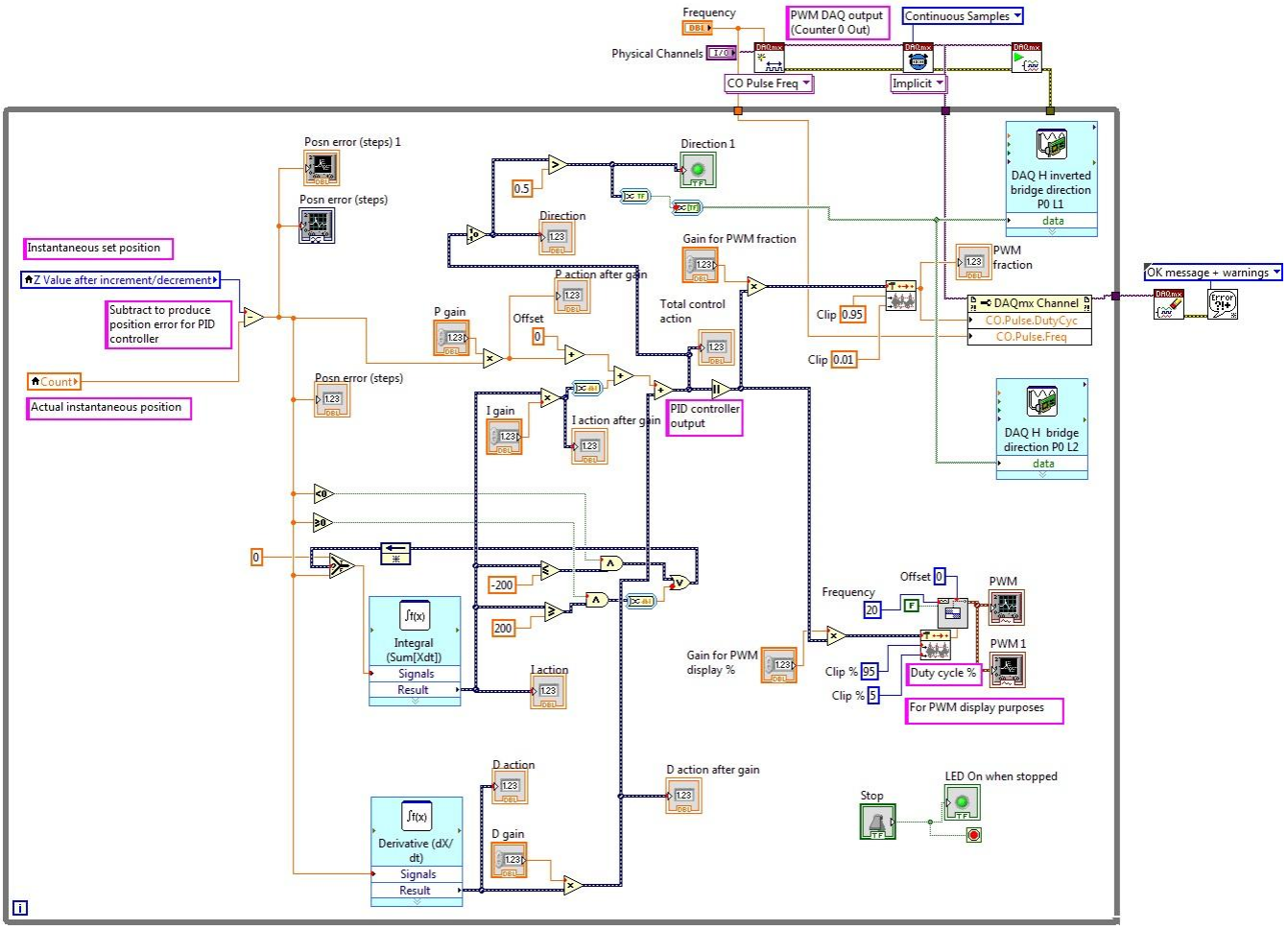


Figure 6.46 Z Axis Position Control Loop Portion of Control Program for Multi-Axis EDM

then a position error will develop between the two axes, i.e. the ratio of travel of the axes will start to differ to the ratio required according to the linear move end point. This is because there are many, many reversals of gap voltage error sign occurring continuously due to the fluctuating gap condition, and a particular sign change may occur just after one axis has incremented, but before the other has (since their shift registers are a-synchronous), and the second register mentioned will then decrement instead, upon its next change. Thus in some instances one register may be incremented (or decremented) a little more than the other, resulting, over time, in a significant position ratio error. This error may of course be limited due to the law of averages, i.e. as many times as one axis happens to increment before a gap error sign change, and the other only after, there may be other times when the reverse happens. But one cannot be sure of this, especially since the registers are not taking the same number of steps during the whole process (unless a 45° angle is being EDM'ed), thus there may be statistically valid reasons for a cumulative error to occur.

6.7.2 Multi-Axis Control: Shift Register Increment Rate Proportional to Gap Voltage Error

An improvement on the control philosophy in the previous section, is to make the period for each shift register not based on a fixed combined axis velocity (i.e. a fixed combined rate of incrementing of the registers), but instead based on a velocity that changes (proportionally) according to the gap size. I.e. when gap size is nearing zero, meaning the electrode is almost in the correct position at that instant, one does not want the rate of change of the register values to remain a fixed value, but it must rather reduce accordingly, such that (large) overshoot does not unnecessarily occur.

Thus the gap voltage error magnitude, as well as its sign, is used in the diagram shown in the figure below, where the gap voltage error size provides a multiplying factor, relative to a 'standard' velocity, to be applied to the standard velocity at any instant, for use in determining the periods of iteration of each shift register. The 'standard' velocity is referenced to a 'standard' (typically expected) gap voltage error size, so that when a typical error size is instantaneously present, the instantaneous rate of change of the shift register values will be the 'standard' value, but when the gap error is smaller (or larger) than the 'standard' error, the rate of change of the registers will reduce (or increase) proportionately.

The LabVIEW Do…While loop for the outer feedback control loop (gap voltage) of the control program is shown in Figure 6.47 overleaf. The size of the error, as well as the sign, is indicated, and a Local Variable is referenced to each, for use in the shift register modules of the program.

The Z axis shift register loop is shown in Figure 6.48 overleaf. The Y axis register loop would be similar. The controlling inputs to the shift register are the same as those in the previous version of the multi-axis EDM control program. The key difference in the shift register is that now the size of the gap voltage error dictates the rate of increment (or decrement) of the register value. A 'Wait period limiter' function is included such that if the gap voltage error is zero at any instant, the Period of iteration of the loop does not go to infinity (zero gap error would imply zero rate of change of increment). As before, the shift register value at any instant is available for use in the position control loop (inner loop), by means of a Local Variable.

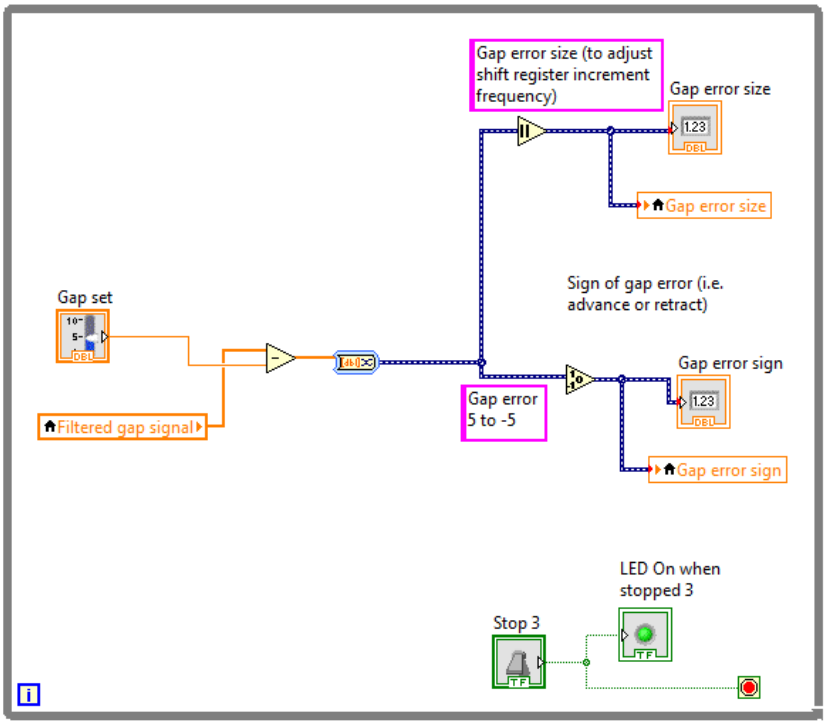


Figure 6.47 Gap Error Sign and Magnitude Determination Loop (Outer Control Loop) for Control Program for Multi-Axis EDM

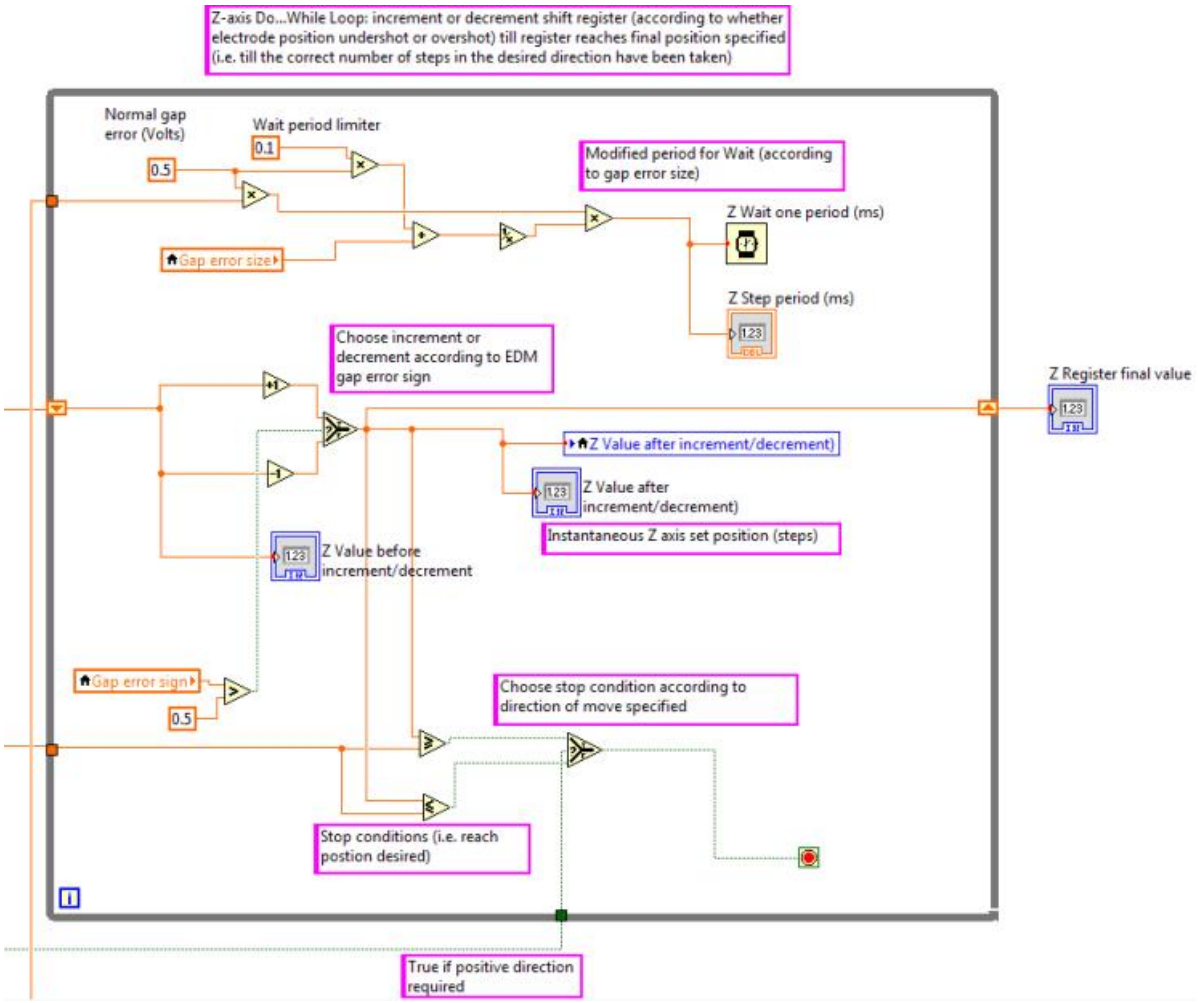


Figure 6.48 Z Axis Shift Register Portion of Control Program for Multi-Axis EDM

This option unfortunately still suffers from the possibility of an accumulative error occurring due to the timing of iterations of each shift register, relative to the (many) changes in gap error sign, which will still occur even though the commanded velocity reduces near cross-over points, making the process more stable.

One of the reasons for making the position shift registers increment or decrement in units (plus one or minus one used in register, dependent on gap error sign), is that each individual shift register 'state change' could be used to produce an output pulse on the DAQ device, for potential use with e.g. a Gecko drive which requires step and direction input pulses (and would then use its own PID loop to produce position changes accordingly). This might have been done for testing two-axis EDM, where one could have employed the linear actuator purchased (with a new H-Bridge driver), and the other one of the existing Gecko drives, serving one of the CNC machine's motors (as per usual). The problem of potential accumulative axes' position ratio error would still however exist.

The method for establishing the actuator's position was discussed briefly above, namely the use of the Count Edges sub-VI. One serious drawback of this method was that if ever the encoder was moved such that the Count value would tend to go below zero, the Count value would instead jump to an extremely large (positive) value, presumably due to underflow of an internal register. Once 'below zero' the Count value did not respond to encoder pulses received, but remained the extremely high value, until the encoder got back to the position corresponding to a zero Count value. This meant that any slight backwards motion of the actuator, if starting from zero, immediately forced a very large error onto the position control loop, and made it saturate completely and force the actuator to full retract (bearing in mind that the Count value was not just incorrect in magnitude, it was also, more importantly, incorrect in sign – thus positive feedback would occur instead of negative feedback, causing control to be completely lost). To some degree this problem could be countered by specifying an initial, positive, Count value (and accordingly specifying the same value for the initial value of the shift register – otherwise a sudden jump would occur at the start of the process, sometimes causing control to be lost due to too large an overshoot).

Due to a variety of persistent problems using the Count Edges encoder counting method just described, later a different encoder counting VI, 'Encoder Position' (a standard VI for quadrature encoders) was utilized instead. [This method is also discussed in the (earlier) section 'Implementing Velocity Control' for Single-Axis EDM.] This counting method did not require the encoder pulses to be decoded first into Pulse and Direction signals, but decoded the encoder signals directly in LabVIEW, such that a Position (output) variable was available from the VI, with encoder direction already taken into account by the VI. Again the VI was inserted in the main control program, and a Local Variable was used to transfer the Position value to the position control loop (inner loop), for comparison with instantaneous position commands from the shift registers, to produce a position error.

One of the reasons for using the previous counting option, i.e. Count Edges, originally, was that it was desired at that stage to have the encoder pulses' frequency (as measured by a DAQ counter input) available for determining actuator velocity, for use in a velocity control feedback loop. This meant that the encoder signals needed to be decoded into a Pulse (state change) signal, and a Direction signal, thus the decoder IC and D-Type flip-flop latch had been used. Thus, a decoded signal was already available for use in determining an encoder Count, defining position, so the Count Edges VI had been used.

If actuator velocity was desired when using the Encoder Position VI counting method, where no decoding was done externally to the DAQ device, this would require that the encoder Position value be differentiated in LabVIEW to obtain velocity (and the sign of the differentiated value would indicate the direction of movement). Alternatively, if the decoding hardware had been maintained and used in conjunction with the Encoder Position VI (for which the encoder signals would bypass the decoder IC), then the frequency of the encoder pulses could still have been used to find velocity (utilizing a DAQ counter input configured accordingly, i.e. for direct frequency measurement, and using the decoder IC), as opposed to differentiating the Position variable; but this would have required more DAQ counter inputs to be utilized, and they were limited, especially since a counter output was required for the creation of the PWM signal for the H-Bridge driver for the actuator (counter ports can be used as inputs or outputs, depending on how they are configured in LabVIEW). If the Count Edges VI had been used to establish position, note too that if a knowledge of velocity had been desired, the Count variable

could be differentiated, and this would give Frequency, without the need for an independent DAQ input frequency measurement. Frequency is of course directly related to velocity, thus actuator velocity would be established.

Once it was decided that the Encoder Position VI be used, this meant that velocity would be established by differentiation of the Position variable. Using encoder pulse frequency, measured directly by a DAQ counter input, was thought originally to be likely more accurate and easier to implement than differentiating a Count value, thus the hardware decoder option had been favored. But there had been concerns anyway as to the accuracy of this method, due to the appearance of the graphs when decoder tests were done. Thus it was felt that a differentiating method was perhaps no worse than using frequency measurement, anyway. Depending on the 'dt' time base of a differentiation function, it was thought that a differentiated encoder velocity value may be 'spikey' and not smooth, as the Count value would be changing a-synchronously to the time base of the differentiating function. However, it was also realized that applying a LabVIEW filter function to the differentiated signal, could probably remedy this, if it was the case.

6.7.3 Multi-Axis Control: Improved (Final) Version, Including Velocity Control Loop

The final control version implemented involved two improvements over the previous shift register method discussed in section 6.7.2 above. Firstly, it was decided to implement a velocity control aspect to the control process, not just position control. I.e. a velocity error was to contribute to the total control action of the LabVIEW controller. Secondly, instead of varying the period of iteration of each axis' shift register, a fixed increment period was used (the same for both axes), but the amount of increment or decrement was made proportional to the gap error (or its representation). Naturally the ratio of movement of each axis, for the linear move, was still taken into account, i.e. the amounts would be different for the two axes, unless a 45° angle move was being undertaken. The modified portions of the LabVIEW block diagram program are shown in Figure 6.49 overleaf, and Figure 6.50 on the page following.

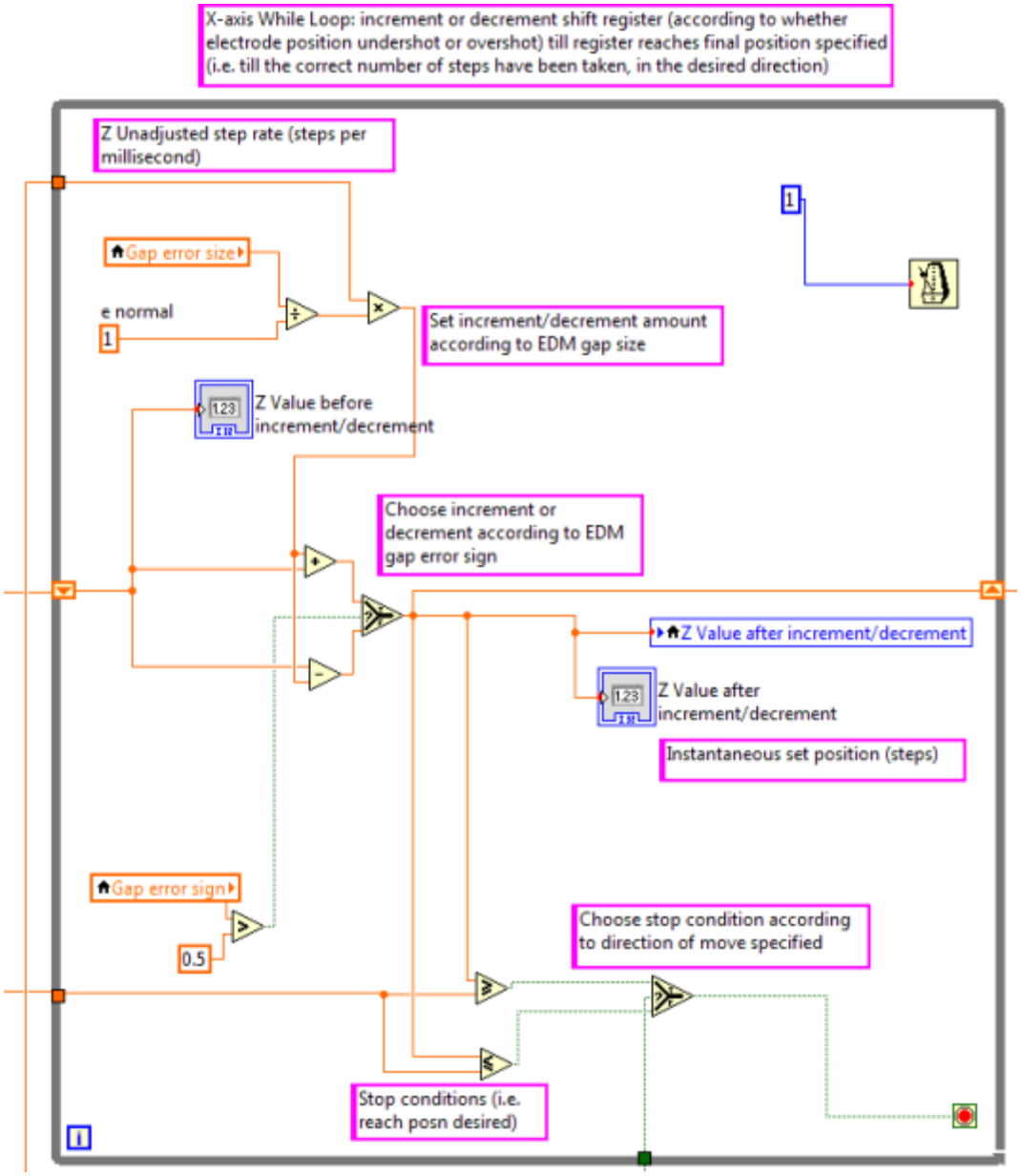


Figure 6.49 Final Z Axis Position Command Shift Register Portion of Control Program for Multi-Axis EDM

Velocity control has been discussed in section 6.6.2 earlier (in the context of single-axis EDM), where it was noted that it makes the process more responsive. For a velocity control aspect to be included in the multi-axis context, it must be implemented in such a way that it influences the responsiveness of the control system, but does not affect the path that the two axes travel during the specified move, i.e. does not affect the axes' movement ratio.

The changing value of a shift register can be seen as a commanded instantaneous velocity (i.e. the rate of change of the commanded position setpoints). If the controller

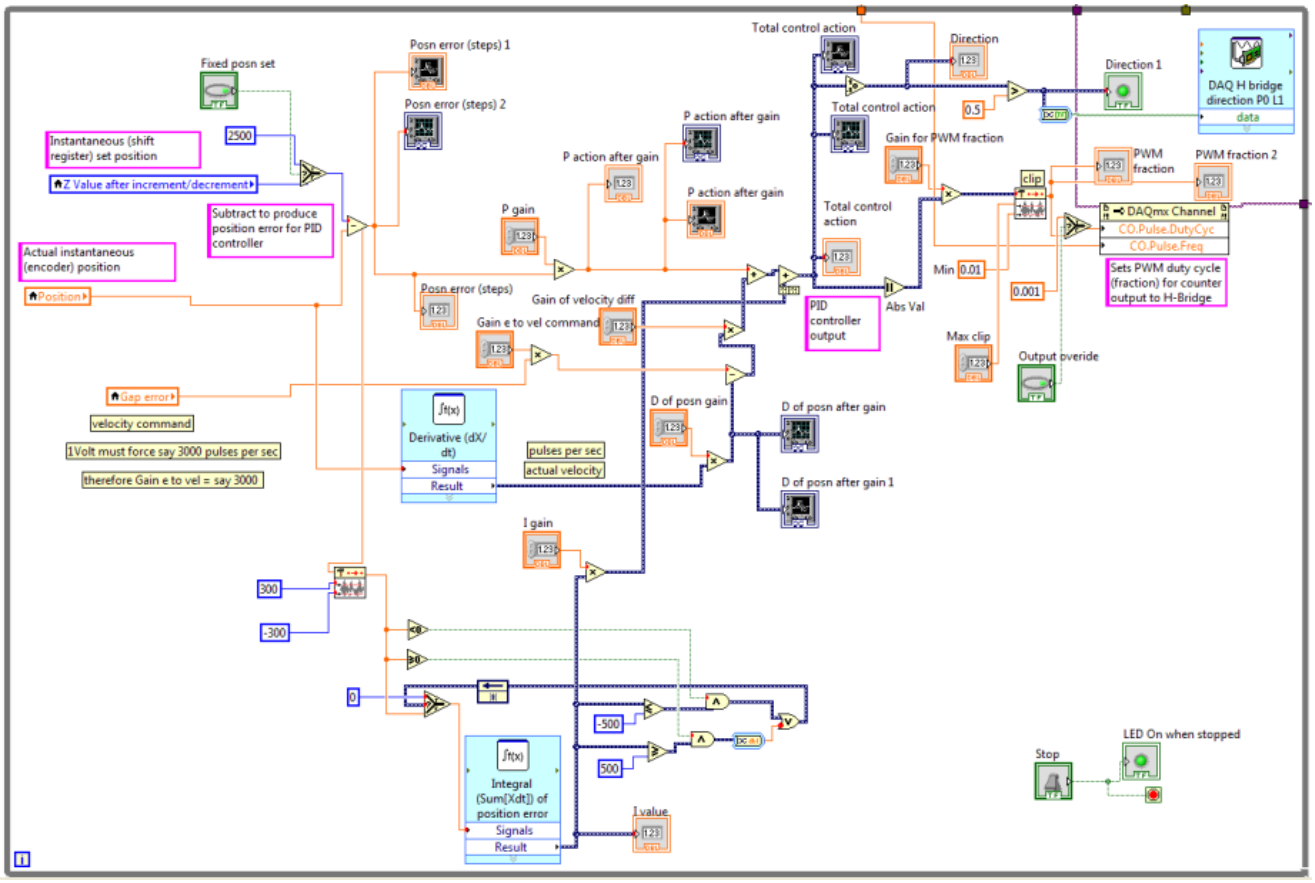


Figure 6.50 Final Z Axis Inner Control Loop Portion of Control Program for Multi-Axis EDM (Velocity and Position, and Position Integrated)

can take action according to this velocity, i.e. make the actuator try to follow/match it, then a more responsive system can be achieved. Again this is because the system does not have to 'wait' for a build-up in position error to occur, to create control action, but can respond almost instantaneously to changes in specified velocity (and velocity changes can be specified almost instantaneously, e.g. if gap distance changes almost instantaneously due to sudden ejection of debris from the gap, or any other reason). In this sense, the position control loop (inner loop) control action can respond to a step change in commanded velocity, if velocity control is present, since one can have a velocity error producing a proportional control action (so step in velocity error means step in control action (PWM output)). If (only) position control is implemented, the control action can only respond to a step change in velocity, over time, as the velocity error integrates to produce a position error (thus control action will be a ramp function, not a step function, i.e. slower response). Now since velocity is the rate of change of a shift register's value, differentiating the shift register value should produce a velocity command for the system to attempt to follow. This is employed in the particular block diagram shown, i.e. Figure 6.50. Alternatively, since the shift register is incremented or

decremented in proportion to the size of the gap error anyway, the gap error is really dictating the rate of change of the shift register, i.e. the velocity command, thus a velocity command can be simply seen as directly proportional to the gap error (so no need to integrate a signal only to differentiate it again to get back to what one started with). Thus, since this introduces less error and is simpler to implement, a velocity command can and was used in the inner control loop, directly proportional to the gap error, with appropriate gain applied. In the simulation environment, the gap error remains represented by the Z axis position error.

However, the control action maintained a component dependent on position (position error), to ensure that the correct trajectory would ultimately be followed. Considering the LabVIEW function block diagram of Figure 6.49 previously, one can see that the velocity error effect is added to the position error effect. A Transfer Function (Laplace) block diagram of this arrangement is shown in Chapter 7. To ensure that the position error ultimately dominates the trajectory, an integration function was included on the position error portion of the control action (as had been done in some of the single-axis program versions – but there the gap voltage error was being integrated, except in the scenario's where position error was used as a surrogate for gap voltage error).

The inclusion of control action based on the velocity error in the manner described above could also be seen simply as PD control (or PID control) based on position, where 'D' (Derivative) means taking the derivative of the position error (which is setpoint position minus actual position) with respect to time – and control action is implemented proportional to this derivative. This is because producing control action proportional to the derivative of the position error (setpoint minus actual) is the same as producing action based on the difference between the derivative of the position setpoint (i.e. a setpoint 'velocity'), and the actual velocity (since the actual velocity is the derivative of the actual position). This scenario is however a little different to the velocity control scenario discussed in section 6.62 previously, since there the gap voltage error dictates the velocity commanded (not a position error or derivative of position error). Also, there no derivative of the setpoint (gap voltage) is taken, and there is no P (Proportional) control.

Finally it could be noted, though, that in the case of multi-axis control incorporating velocity control, since the shift register increment rate was set proportional to the

instantaneous gap voltage error, there is again a similarity with the velocity control scenario of section 6.62, as in effect the gap voltage error is dictating the commanded instantaneous velocity, as it was in section 6.62. But this only occurs due to the decision to make the shift register increment rates proportional to gap voltage error. Also, in the present case, P control action is also present, whereas in section 6.62 only velocity control was present.

Other improvements of the final version over the previous version is the inclusion of a 'Fixed position override' where the user can select that actuator control be to a fixed position, for stability testing purposes (i.e. shift register is bypassed), and an 'Output override', such that the PWM duty cycle of the actuator can be set to a very small value (near zero), for safety purposes during testing i.e. when control is not achieved.

The LabVIEW front panel for the program is shown in Figure 6.51 overleaf (a little too reduced to be seen clearly). A variety of parameters are displayed on the front panel, and the user can set various parameters such as gains, Z and Y distance to travel, filter cut-off frequencies, etc. The abovementioned overrides can be selected. Pertinent parameters are also displayed in graph form, e.g. gap voltage, encoder position, etc.

In Figures 6.52 and 6.53 on the page following, printscreens of portions of the front panel of the program are shown, captured while EDM was being performed using the linear actuator, mounted on the CNC machine. It can be seen on the second portion that the EDM gap signal being used was quite noisy, but this is due to the selection of the relatively high cut-off frequency for the filter (195 Hz), at the time. Stable EDM was achieved, although the position error was at times unacceptably high. This is discussed later. While using the program, stable EDM continued for long periods of time (unlike the method where EDM used the CNC servo motors, together with Mach-3, where there were clear breaks (stoppages) in EDM, every few seconds). The process was also much more stable than the method employing the CNC servo motors, but driven by the EDM machine's own controller. In that instance, although EDM was generally continuous, the (average) gap voltage seen on the EDM machine's panel fluctuated widely, continuously. Whereas with the linear actuator, the gap voltage seen on the EDM machine's panel was quite stable, at times barely moving. Figure 6.54 on page 125 shows the gap voltage stable at about 55V.

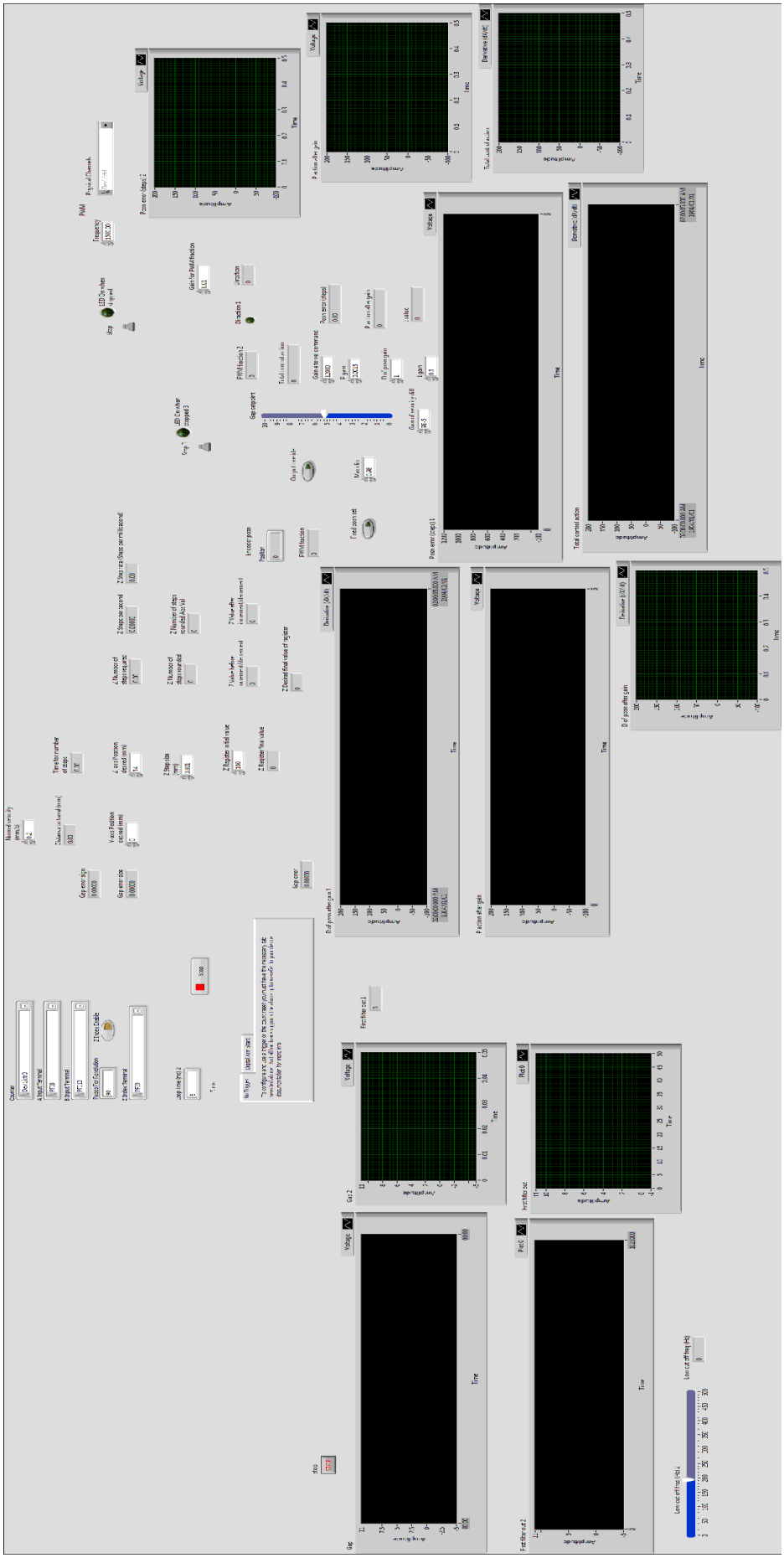


Figure 6.51 Final Control Program Front Panel for Multi-Axis EDM

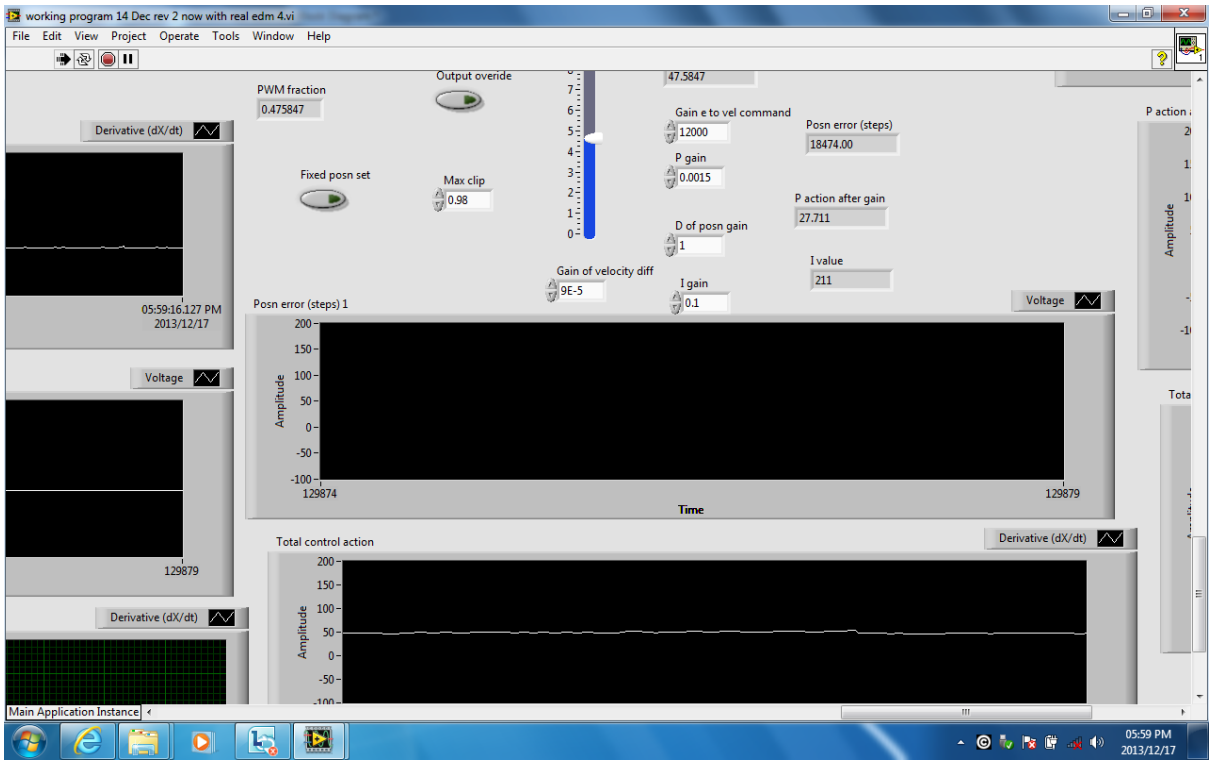


Figure 6.52 Portion of Front Panel Captured during EDM Process using Linear Actuator

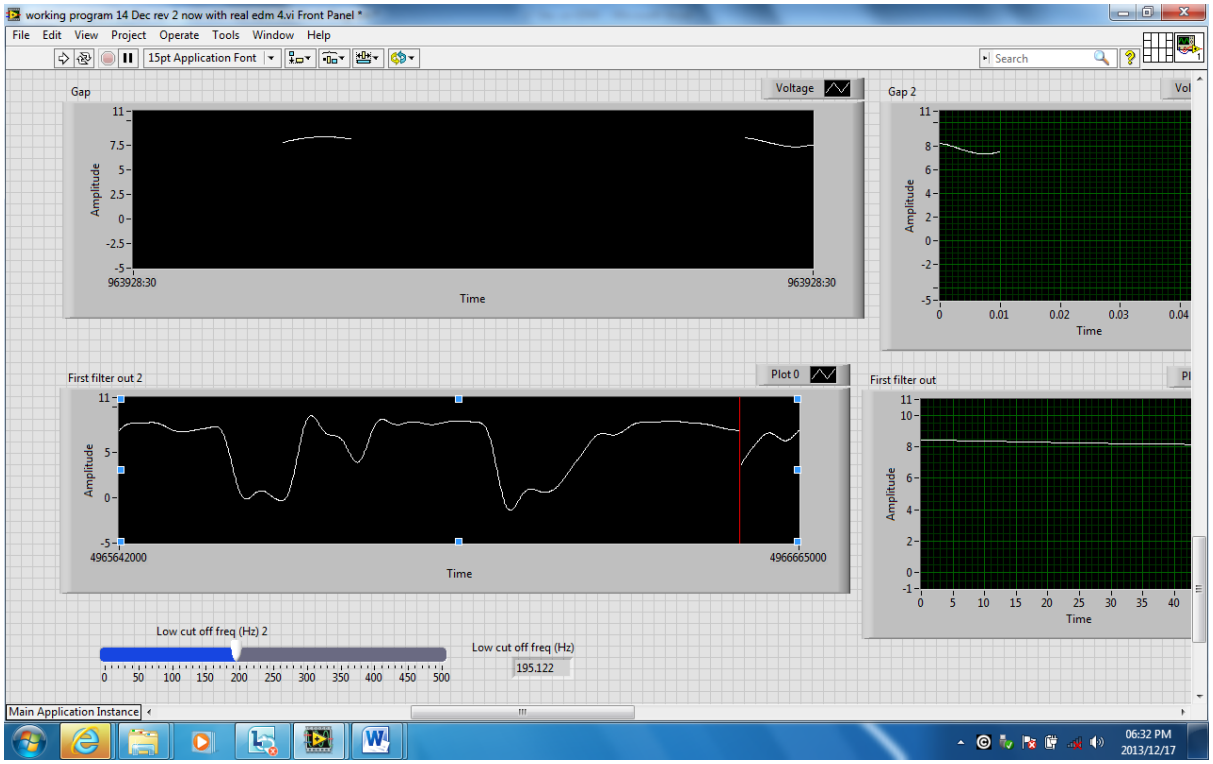


Figure 6.53 Portion of Front Panel Captured during EDM Process using Linear Actuator

Figure 6.54 Portion of Alic-1 Control Panel Captured during EDM Process using Linear Actuator, Showing Stable Gap Voltage of 55V

6.8 Modelling of EDM Process in LabVIEW

Some versions of the shift register-based cascade control scheme used the instantaneous position of the Z axis actuator (i.e. its encoder position) as surrogate for EDM gap voltage, for testing purposes, mainly so that development could take place remotely from the CNC and EDM machines; also it was easier to test programs with the actuator accessible near the DAQ and PC (lap-top) running LabVIEW, because if control was poor and the actuator was tending to oscillate violently, then manual intervention (e.g. stabilizing the actuator's movement with one's hand by creating some damping) could be taken; this would also prevent damage to the actuator occurring. It was felt better to have stable, working programs, before implementing them in a real EDM process on the CNC machine tool.

It may seem counter-intuitive to use one of the outputs of a control system (Z axis position) as an input to the same control program. However, if one considers that this is really the same as using the LVDT position sensor as surrogate (only the signal is digital not analogue), then it may make more sense. Also, when noise is added as described in the following paragraph, it becomes a realistic signal.

In order to use Z axis actuator position as surrogate for EDM gap voltage, it was necessary to apply artificial 'noise' to the position signal, to make the control process

more realistic for testing purposes (naturally it is easier to obtain stability with a ‘clean’, smooth, position signal, than a noisy, fluctuating EDM gap voltage signal). Also, to force the position setpoint to (gradually) move (as it would in EDM since the workpiece material would be slowly eroded), and not just (potentially) oscillate due to under- and overshooting, it was necessary to force (add) a (relatively) gradual position increasing signal to the process setpoint, such that eventually the end point of a particular linear travel (according to specified Z and Y end point values in the user interface), was reached. In other words there needed to be an overall tendency towards the process end point, while on a shorter time scale oscillatory movement was naturally occurring, according to the nature of the control process, where a moving, noisy position setpoint was being followed by the position feedback loop. Thus a simulation technique of a sort was being implemented, but in conjunction with a real actuator and H-Bridge driver. Note that when ‘increasing signal’ is mentioned above, this refers to an increase in the shift register values (in the Z axis at least), as the program was arranged such that increasing shift register values meant advancing electrode, which is naturally downwards for EDM.

To add noise to the surrogate EDM gap voltage signal (i.e. the Z axis encoder position signal), a noise generating LabVIEW function was applied, with suitable parameters in terms of (estimated) amplitude and time scale of signal. Adding noise in turn creates more natural ‘noise’ in the process, in that the actuator(s) does not have time to settle on a particular position value (meaning the surrogate setpoint does not settle on a stable value), so oscillating, changing conditions are enforced, which is much more representative of the true EDM process. The fluctuating conditions are seen as rapid changes in increment vs. decrement of each shift register, i.e. each shift register’s value fluctuates around some value which is the current average setpoint of the position command feeding into the position control loop for each axis, i.e. inner loop, which dictates PWM output to each H-Bridge. The fluctuations are seen also in the position loops’ outputs as the control loops ‘chase’ the fluctuating position commands.

It was felt justified to use only the Z axis encoder position as surrogate for EDM gap condition, as the travel of the two axes are directly related – similar fluctuations would be seen in each, as they essentially move together (the sign of the instantaneous gap error dictates the increment/decrement status of both shift registers, simultaneously, both in real EDM and in the simulation mode). One shift register may of course be incrementing

while the other is decrementing (Z travel would most likely be positive (downwards), and Y axis could be positive or negative (left or right), depending on the end point specified for the travel); but the sign difference would be maintained at all times during the process, i.e. it is consistent so the two axes can still be seen to be moving ‘together’.

To create the general tendency for the process to move towards the end points of the travel specified (simulating erosion of the workpiece), the following philosophy was implemented.

The true rate of workpiece erosion is dependent on the quality of the EDM gap, i.e. whether its size is stable and at the desired distance from the workpiece (both of these factors meaning that minimal short circuits and open circuits are occurring, during which no material removal takes place). Thus for simulation purposes a ‘workpiece surface height’ (WSH) function was implemented, based on time and a ‘workpiece removal rate’, which in turn is determined in the function by the quality of the gap, for which the gap voltage error would be representative (smaller error means better gap quality). And since the gap voltage error is represented by a position error (the better the position loop follows the position commands, the smaller the error, both in the simulation and in true EDM), the gap position error was used in the function (a smaller error meaning a faster material removal rate because of the belter quality of the simulated gap condition). Workpiece surface height (WSH) is defined by the starting point (taken as zero), plus the distance of erosion of the material, which is the integral of time and workpiece removal rate (positive meaning advance electrode, i.e. surface height is defined as positive downwards). Removal rate in turn was made inversely proportional to gap error size (position error as surrogate), as a smaller error represents a better quality gap, hence faster material removal rate.

It was important to use the magnitude only of the position error, as the error would appear approximately zero over time if its sign was included, due to averaging (approximately as much time is spent with a positive gap error, as with a negative gap error, as the process fluctuates). Ideally the steady state portion of the gap error should be excluded from ‘gap error size’, for use in the formula, because if the gap error is steady, then good EDM will be occurring, hence high removal will be occurring, albeit at a slightly different surface finish than desired. However for simplicity, just the ‘gap error size’ was used. Thus the formula used was as follows.

Workpiece surface height (EDM erosion) effect (surface height defined positive down):

WSH = Initial Height + Integral of [Erosion Rate] with respect to time

Now ideal (zero gap error) erosion rate is defined as ‘Constant’; but erosion rate gets reduced if gap error is large; therefore an amount of (k x Gap error size) is subtracted from Constant, to produce ‘true’ rate of erosion, where k is a suitable mutilplier.

Thus

Erosion Rate = Constant – k x Gap error size

Now an amount of erosion is just the rate multiplied by a time period, therefore

WSH = Initial height + Integral of [(Constant - k x Gap error size)] with respect to time

In LabVIEW, the Integration function is produced by using a shift register for the WSH variable, where a small ‘delta Time’ interval is used in a continual summing, as the shift register loop iterates. The Wait function is used in the shift register, to dictate the iteration rate (i.e. to dictate the delta Time period).

If however the Erosion Rate works out negative at a certain instant this would imply ‘upwards’ EDM (for the Z axis at least), thus Erosion Rate is made zero for delta Time instances where Erosion Rate would be less than zero (i.e. use Zero x delta Time for time periods where (Constant - k x Gap error size) < 0). In other words, the ‘worst’ rate of erosion is just zero, because if gap condition is really bad, simply no erosion will occur; but surface will not become higher! (‘higher’ here meaning really higher, not positive downwards).

The WSH effect and Noise is added (in a separate Do…While loop for the outer control loop, and by means of a Local Variable) to the position feedback to produce an ‘effective’ position, which then gets compared with the gap voltage setpoint, to dictate the rate of increment or decrement (according to sign) of the shift registers which are producing the position commands for the inner feedback control loop. In this way, the position shift registers will eventually be incremented to their final values, and the linear actuator will

eventually move through its whole range specified according to the move amount originally set by the user.

The implementation of the WSH function in LabVIEW is shown in the Figure 6.55 and 6.56 below. In this particular instance, the LVDT position signal is used in the feedback path, but (Z axis) encoder position could also be used.

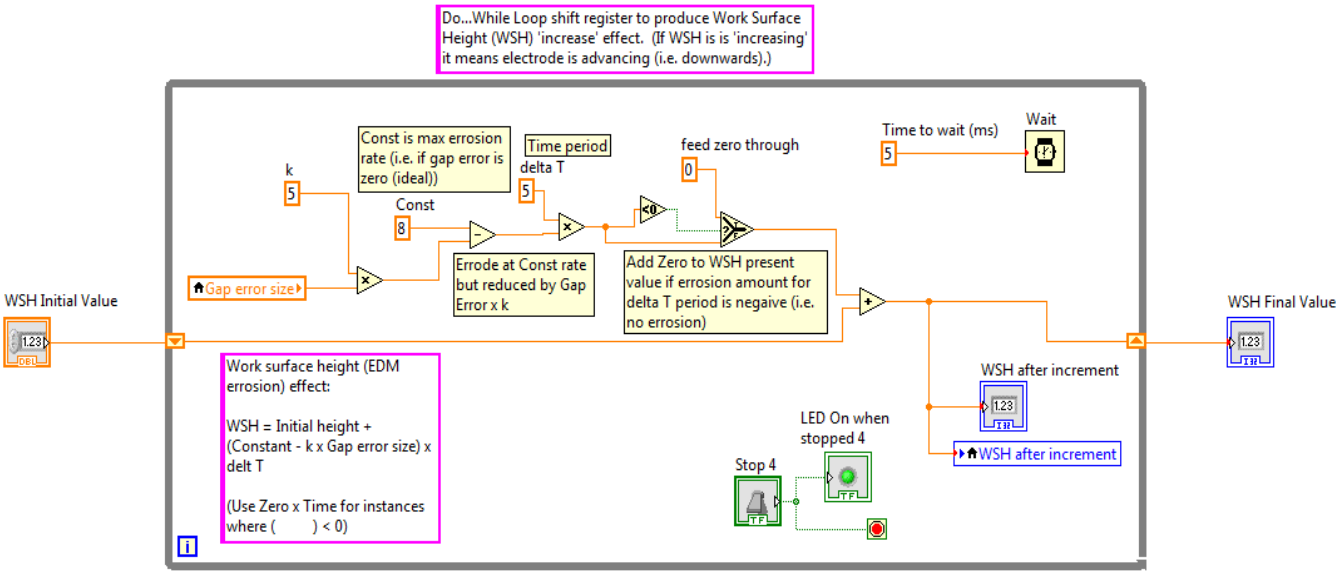


Figure 6.55 Workpiece Surface Height (WSH) Effect (Workpiece Erosion Effect) for EDM Process Modelling

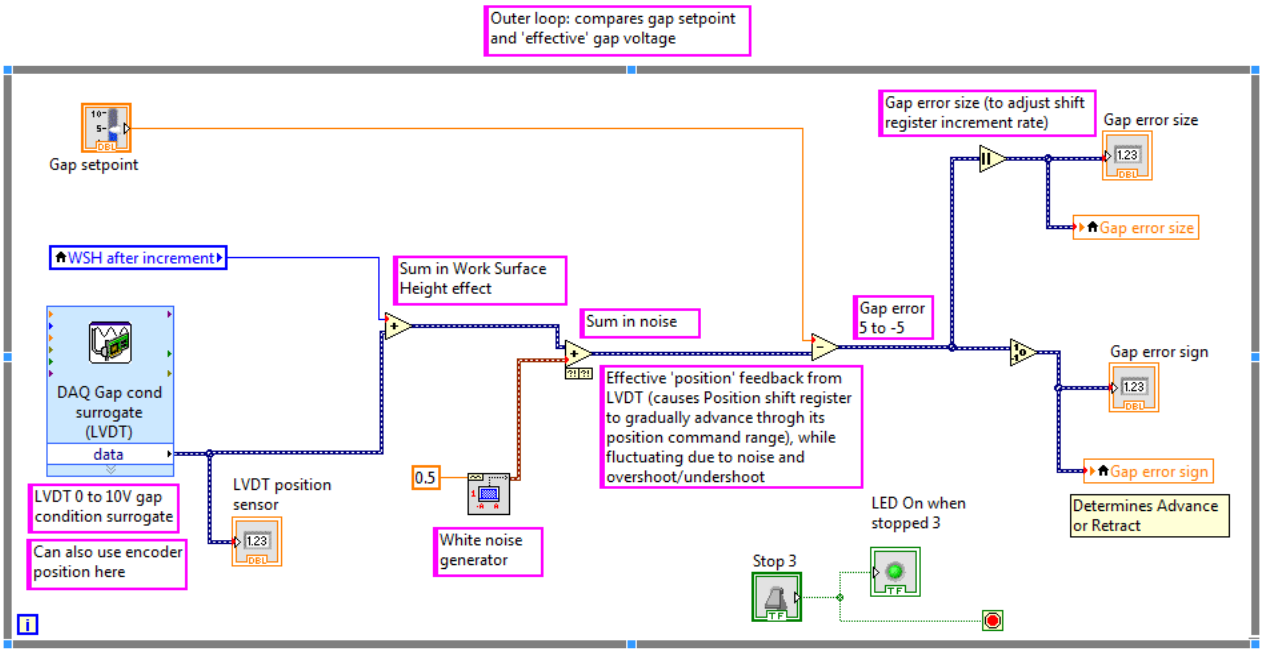


Figure 6.56 Outer Control Loop: Position Feedback Adjusted by WSH and Noise, to Simulate EDM Process

By using this method of creating a slowly moving average position setpoint, the quality of the process (essentially the stability of the process, since more stable means less gap error) can be, to a fair degree, assessed simply by observing the time taken to traverse a particular move specified. Naturally a graph of error vs. time would do much the same thing, as would a variable showing the average error (again only error magnitude used in the determination of this variable), and these could be used to evaluate the influence of, for example, certain controller gains (K_P, K_I, etc.), or any other parameter that can be adjusted. Using the workpiece surface height function conveniently means that a simulated EDM process can be 'sped up' simply by changing the coefficients used in the function. Thus one can see the influence of changes in controller parameters, quickly.

6.9 Conclusion to Chapter 6

In this chapter the implementation of a variety of integration schemes utilizing an external controller (implemented in LabVIEW, making use of a data acquisition device), combined with the use of a linear actuator, is described in detail. Simplified control block diagrams of the various arrangements are provided, along with more realistic control implementation diagrams. LabVIEW programs are shown and explained, and the results of the various types of control noted and discussed. A method of modeling the EDM erosion rate for use in simulation of the EDM process as a whole, including noise injection, controller, servo drive, and motor, is also given.

Satisfactory EDM was achieved in some of these arrangements. An important observation was that the inclusion of a velocity feedback loop in the motor controller improved stability significantly. Although programs for the implementation of multi-axis EDM'ing were developed, they could not be tested due to hardware limitations.

Chapter 7 following investigates the development of a simulation model for the entire EDM process. Most important though is the modelling of the actuator and controller, in that the effect of various parameters can be observed in simulation, so that the real process controller could be optimized.

7. Modelling of EDM Control System in Scilab

As part of the research project, an investigation into modeling the system in Scilab was done. Scilab is a program very similar to Matlab (but is freely available), and has an environment called Xcos for modeling and simulation of control systems, similar to the Simulink environment in Matlab. Various portions are available for the representation of a control system, e.g. Pole-Zero method, Transfer Function method, State-Space method, etc. In this instance the Transfer Function (Laplace) was used. The main purpose was to model the actuator and controller, in order to be able to compare the model to the real implementations, and see the effect of modifying various parameters or of using different types of controller, so that the process could be optimized. If the EDM process itself could be (simply) modeled, that would be advantageous.

7.1 Scilab Models and Simulation Results

A number of different models of control philosophies, each basically representing a version of a control method implemented in a LabVIEW program, were modeled and simulated in Scilab. Naturally the main difference is that the actuator/motor itself is also simulated, whereas for the LabVIEW implementations, the actuator was a physical device. Modeling in Scilab would ideally mean that various control philosophies (and parameters e.g. PID gains) could be simulated and their performance evaluated and compared, such that the system could be optimized. Also, performance could be compared to physical results achieved using the hardware, to verify the Scilab models. The particulars of the models implemented are at this stage somewhat arbitrary (e.g. actuator parameters such as armature resistance, back-emf voltage constant, PID gains, and setpoints), but they nevertheless are representative of the type of system and still enable performances to be compared or control philosophies to be justified. It would only be a matter of carefully making the coefficients (gains etc.) used identical to the LabVIEW implementations, and using the correct actuator parameters, if more realistic performance evaluation is desired. Certain parameters of the actuator were in fact measured (see next section), but the Scilab models have not been adjusted to correlate with the measured parameters. The values of some parameters were however chosen for a reason, for example there are certain multipliers to convert position error to voltage error, which it represents. Also for example gains are applied to convert a gap voltage error into a suitable velocity command in the case of velocity feedback being used. In

general, suitable gains were estimated, applied, and adjusted until a reasonable response (where possible) was seen. The idea of a 'proportional band' was employed for this purpose, too, but in some cases gains had to be changed significantly to achieve good results.

The basic actuator model is taken from [38], who represents a DC motor using Laplace Transfer functions including a feedback loop as explained below. A moving coil linear actuator is identical to a DC motor, except that linear and not rotational motion is produced. Thus there are correlating parameters. For example, the back-emf constant K_B for a DC motor gives armature volts produced per rotational velocity unit (e.g. RPM, or better, rad/sec), whereas the equivalent term gives volts produced per linear velocity (e.g. m/s). Likewise, the torque constant K_T, usually given in N.m per amp, is equivalent to a force constant, with units of N per amp. Armature resistance and inductance have the same definitions for both a motor and the linear device. Rotational damping (viscous friction) for a motor usually has units of N.m per rad/sec, whereas the equivalent would be N per m/s. 'Spring' stiffness k in the rotational environment would have units of N.m/rad, the equivalent being N/m in the translational environment. Angular acceleration alpha (Rad/s^2) is equivalent to linear acceleration (m/s^2). And naturally mass moment of inertia is equivalent to mass.

One source of error that can arise when modeling a motor/actuator is that the applied voltage is usually not a 'smooth' analogue value, but rather a PWM function. The effect of armature inductance would be presumably far more significant where a PWM amplifier is being used, as a sharply fluctuating voltage is being applied to the armature, whereas if voltage is relatively smooth, the impedance produced by inductance would be far less since a voltage drop only appears across an inductor when there is a tendency for current to change (no voltage appears across an inductor when there is no change in current since V=Ldi/dt for an inductor). [39] provides a model for the effect of applying a PWM driving voltage when inductance is involved, but for the purpose of the present report, applied motor voltage is treated as if it was a smooth analogue value (the average value of the PWM function at any instant).

7.1.1 Complete Model of Linear Actuator and EDM Process Including Workpiece Erosion

The particular Scilab representation of the system in Figure 7.1 below represents the system as developed in the LabVIEW program which includes a shift register (essentially doing the job of a counter (integrator)), as well as a function representing the EDM process itself (i.e. workpiece surface height reduction due to erosion), implemented in the LabVIEW program by an appropriate continuous amending of the gap voltage error signal (or the overall position error, where position is being used as a surrogate for the EDM gap error), based on the advance of time, taking into account the instantaneous value of the magnitude of the gap error (such that erosion rate is slowed when error is large, which is true to an EDM process). This arrangement is shown in section 6.8, in the LabVIEW implementation.

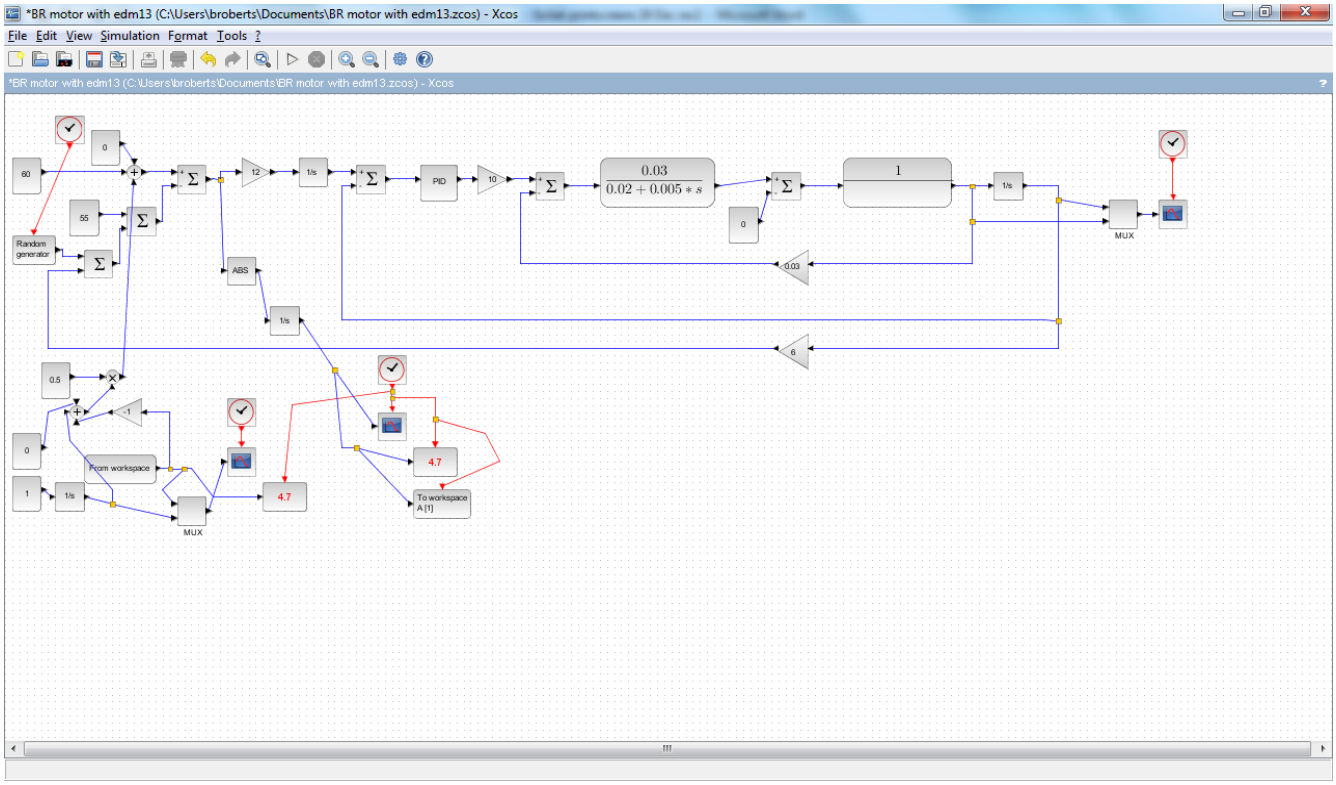


Figure 7.1 Scilab Model of Linear Actuator and EDM Process

Part of the model is repeated in Figure 7.2 overleaf (larger), for clarity:

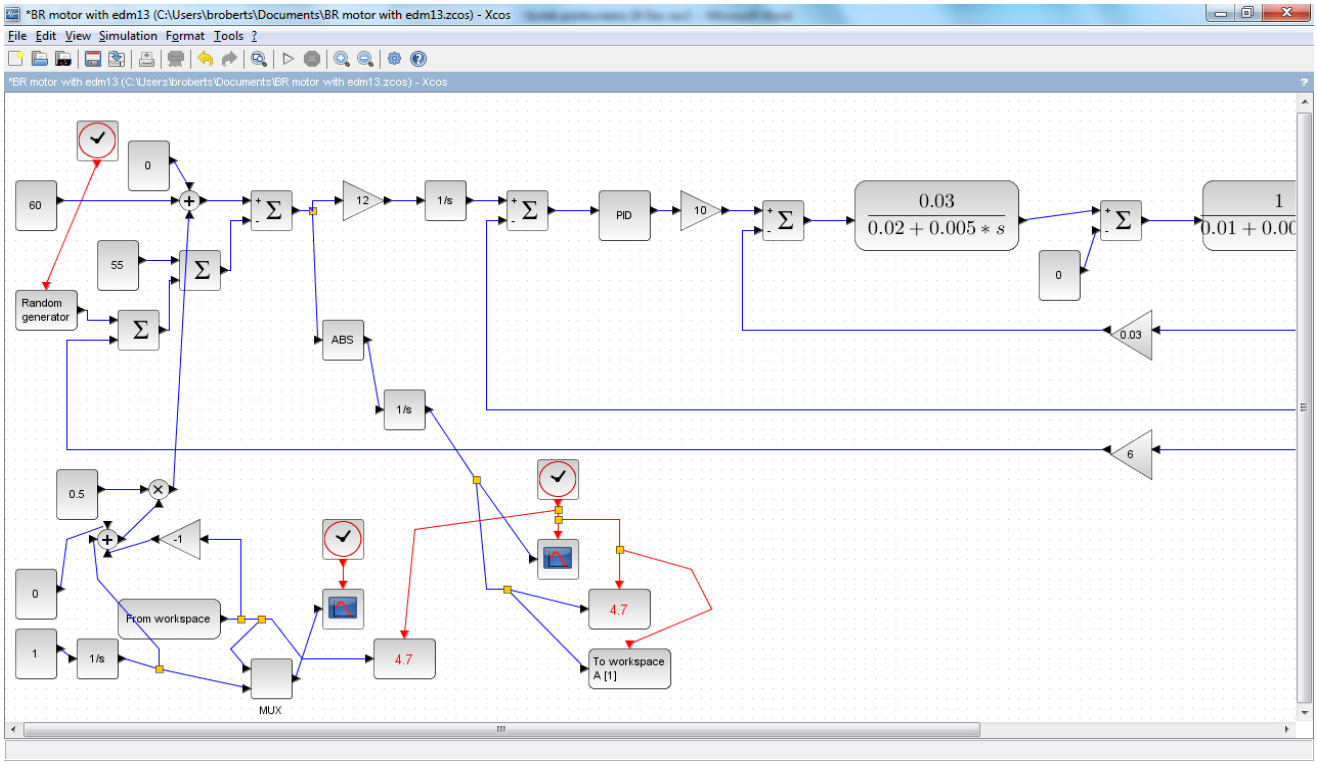


Figure 7.2 Scilab Model of Linear Actuator and EDM Process (Larger)

The inner loop is the actuator/motor itself. The first transfer function block of the inner loop represents the armature resistance, inductance, and the Torque Constant K_i (Force Constant if a linear actuator); the second block represents the armature mass moment of inertia, and the torsional damping (mass and linear damping (viscous friction) for actuator). The motor itself includes a feedback loop due to the nature of a motor having a back-emf (proportional to velocity), which affects the voltage seen by the armature according to the instantaneous speed of the motor – the armature sees less voltage when motor is turning faster (or in the case of a linear actuator, when it is translating faster). One can see that the velocity (after the motor transfer function) is what is fed back in the inner loop (with the feedback path gain being the back-emf constant K_B). Between the two transfer function blocks representing the motor is a summation where a torque disturbance can be added (zero in the present case).

After the velocity output of the motor is a (1/s) Laplace function (integration). After the (1/s) function we naturally have position, since velocity has been integrated (the (1/s) can also be seen, in the digital environment, as a counter counting encoder pulses to measure position; before the (1/s) we have velocity which would be represented by an encoder pulse frequency). The actuator position is fed back in the next loop (middle loop), a position feedback loop, to the Summation sign, where it is compared to a

position setpoint (command) from an outer loop, which represents the EDM process function. After the position error is produced, it is acted on by the PID controller of the middle loop.. The EDM process is represented by, essentially, a ramp function of the setpoint of the outer loop, combined with random noise due to the nature of the EDM process. But it is not a ramp function exactly – there is a fixed, initial value, 60 in this case (with 55 representing the initial position of the actuator), to which is added an amount which increases with time. This amount represents the workpiece surface height gradually changing (eroding, taking downwards as positive). So the control loop, overall, is following this increasing value of 'setpoint', i.e. the moving surface of the EDM process (the actuator's position is following this 'setpoint', since that is the output of the control loop). Part of the amount being added to the initial setpoint, is (0.5·t), where t is time – this is a ramp function, representing ideal EDM, i.e. a 'fast' material removal rate. The 'time' variable is obtained by integrating unity, i.e. (1·1/s) as a Laplace transform; the 0.5 is an arbitrary multiplier. But this ramp is made smaller by (0.5·(integral of $\left|\text{error}\right|$)). This component represents a 'worsening' of the process due the presence of non-ideal gap distance, i.e. error. Thus, the material removal is made slower when error is large, which is true to a real EDM process. Naturally this is a simplification, as the rate of erosion is not necessarily linearly related to error as it is in this model. However, it does demonstrate the principle. Note too that position is again being taken as a surrogate for the EDM gap voltage – the setpoint is a moving surface position, and the control loop output as a whole is the actuator position which is trying to track the moving setpoint. But bearing in mind that a gap voltage is produced by a distance error between workpiece and electrode, it is not really strange to be using a position error to represent gap voltage error.

The 0.5 multiplier shown is an arbitrary 'gain' and applies to the ramp with time and the 'worsening' of the EDM erosion rate due to gap error. Separate multipliers could however be used for the two components – but the program only demonstrates the principle.

The position error is integrated by the (1/s) function before being passed as a command to the (middle) position loop. Without this integration, the process is unable to follow the effective ramp function with steady state error (since the integration value in the PID controller was set at zero in this instance), and the 'EDM process' error is seen to

increase (the integral of the absolute value of error becomes roughly a parabolic function, not a roughly linear function).

The first three graphs (Figures 7.3 – 7-5 below and overleaf) are the output of the Scilab program when it is run. In this case the random noise generator range was set to -0.2 to 0.2. The graphs show a 30 second time period. In the first graph (Figure 7.3), the actuator velocity is seen (Green), together with its position (Black). A large response is initially seen due to the difference between the initial set point (60) and the initial actuator position (55). This sort of large initial error is like a step function, and causes a large initial response, which later reduces and becomes relatively steady (bearing in mind that the random noise is always present, therefore there is continuous control action required). It is evident that although significant control action is occurring in terms of velocity, the position that this velocity is achieving is relatively stable (increasing gradually, representing the workpiece erosion effect). The second graph (Figure 7.4) shows the gradual integration of the error (roughly linear), and the third (Figure 7.5) shows the (fixed) ramp function (Green), together with the integration of the error (Black).

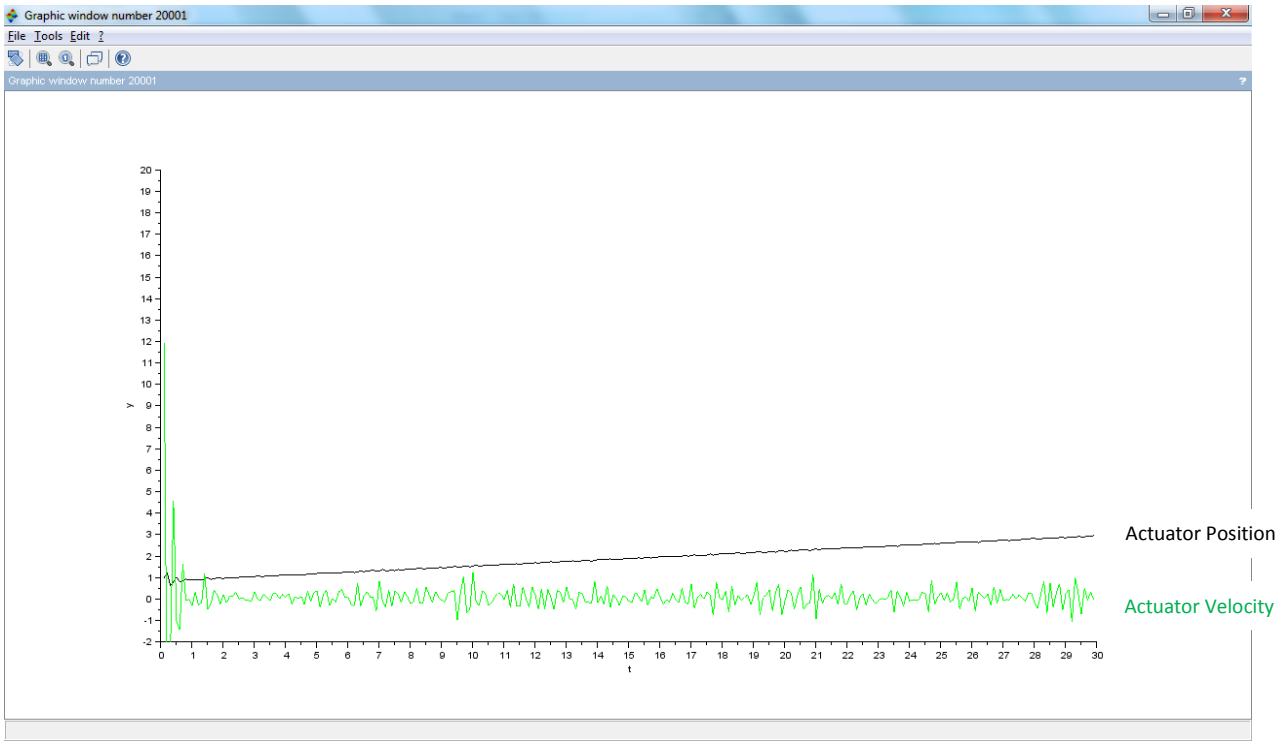


Figure 7.3 Scilab Simulation: Linear Actuator and EDM Process Including Erosion Effect: Actuator Position and Velocity

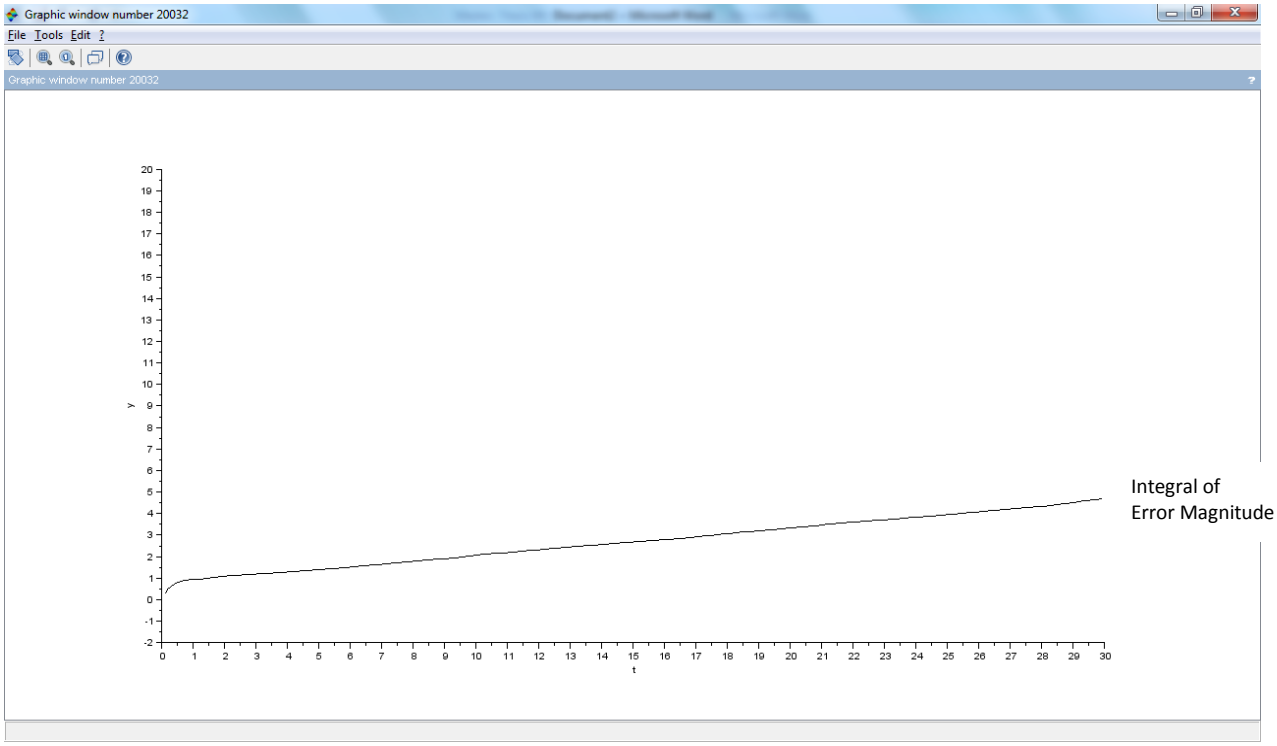


Figure 7.4 Scilab Simulation: Linear Actuator and EDM Process Including Erosion Effect: Integration of Magnitude of Error over Time

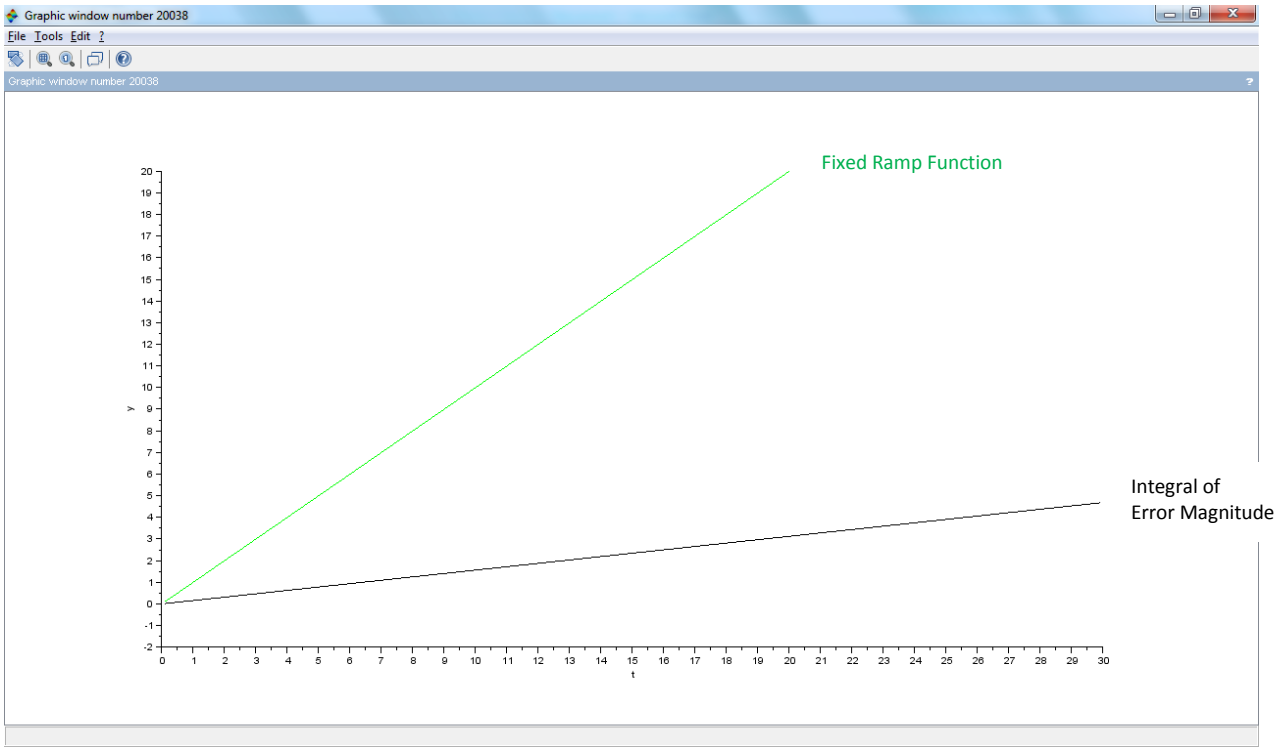


Figure 7.5 Scilab Simulation: Linear Actuator and EDM Process Including Erosion Effect: Fixed Ramp Function, and Integration of Magnitude of Error over Time

The following three graphs (Figures 7.6 – 7.8 below and overleaf) show the same parameters, when the random noise generator range was set to -0.5 to 0.5. The higher error values are visible, and the slower rate of movement of the output (i.e. lower erosion rate) is evident. In addition, significantly higher velocities are seen in terms of the process responding to the higher random noise, i.e. attempting to follow the varying outer loop setpoint.

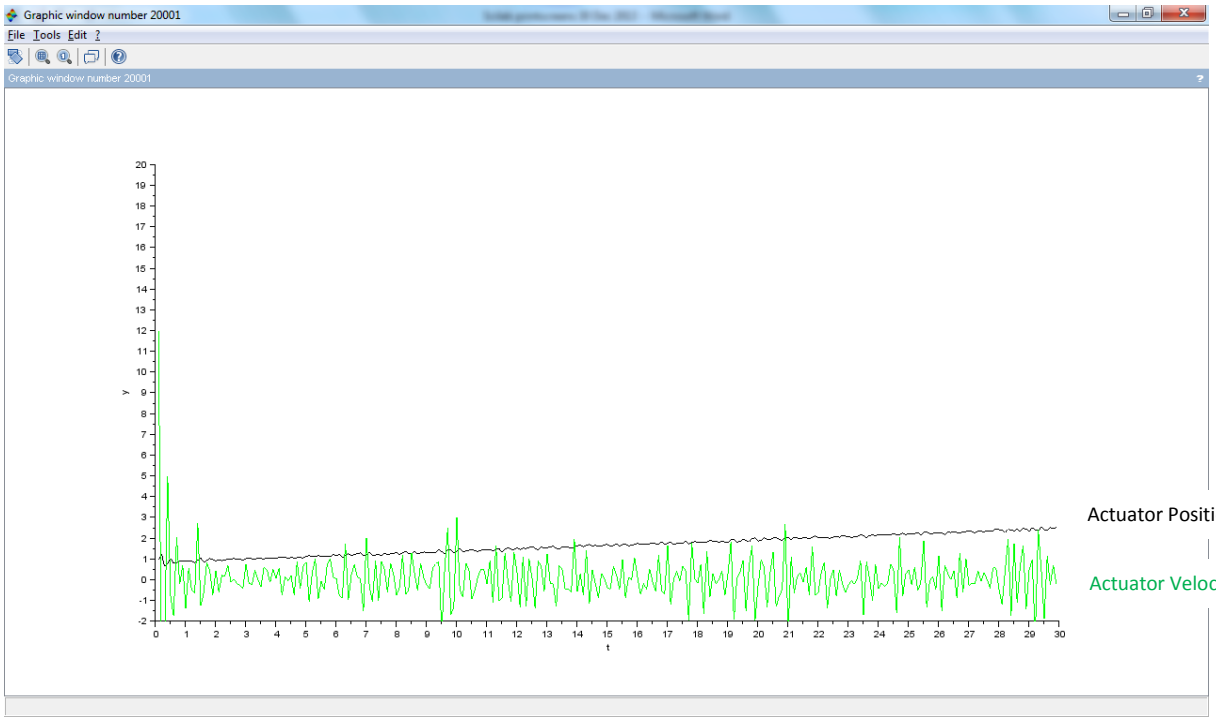


Figure 7.6 As per Equivalent Graph Above, but Random Noise Set to Amplitude of 0.5

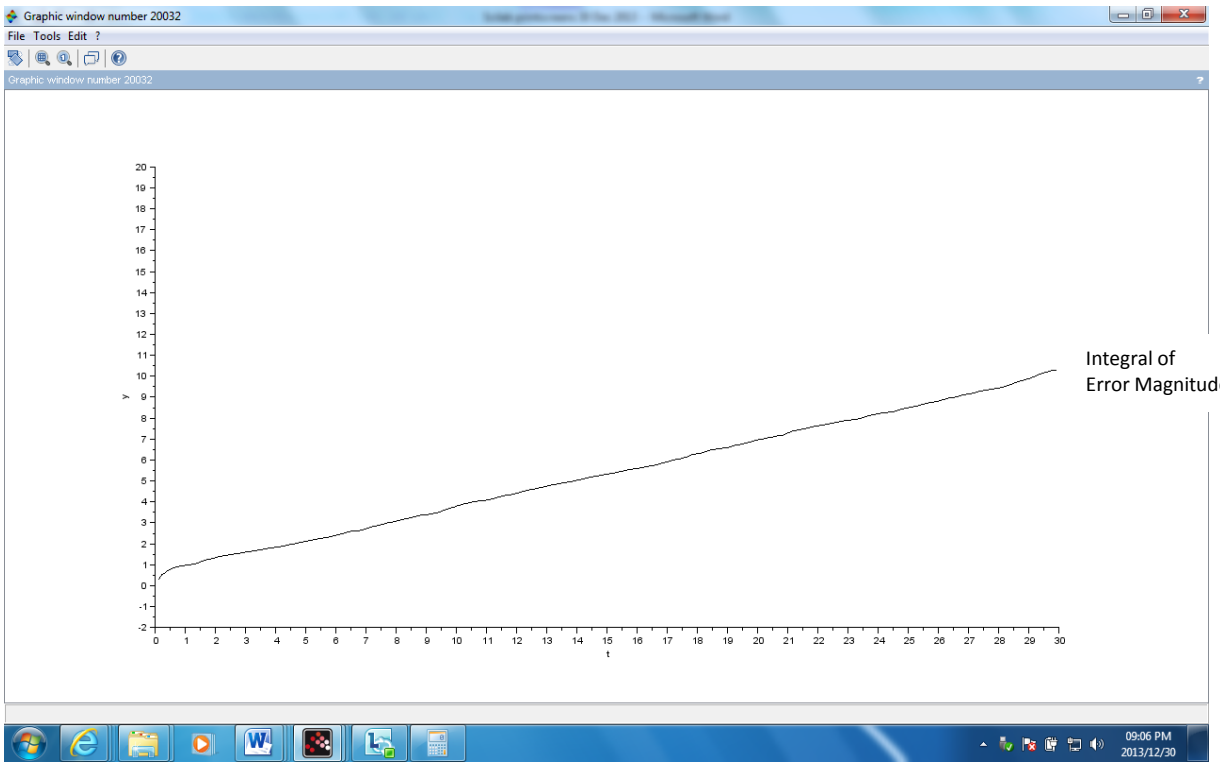


Figure 7.7 As per Equivalent Graph Above, but Random Noise Set to Amplitude of 0.5

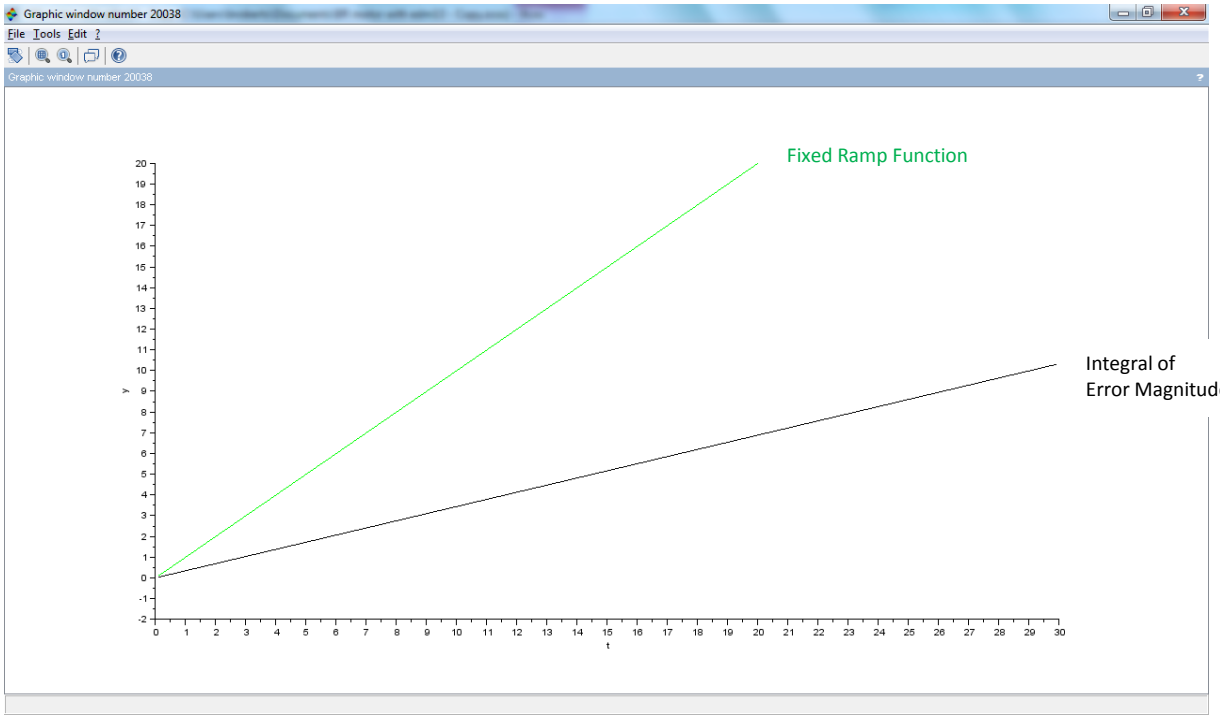


Figure 7.8 As per Equivalent Graph Above, but Random Noise Set to Amplitude of 0.5

The graph of Figure 7.9 below shows in more detail the system response (velocity and position) for the first three seconds. The velocity is seen to stabilize after a larger initial response due to the initial difference between setpoint and position (effectively a step function).

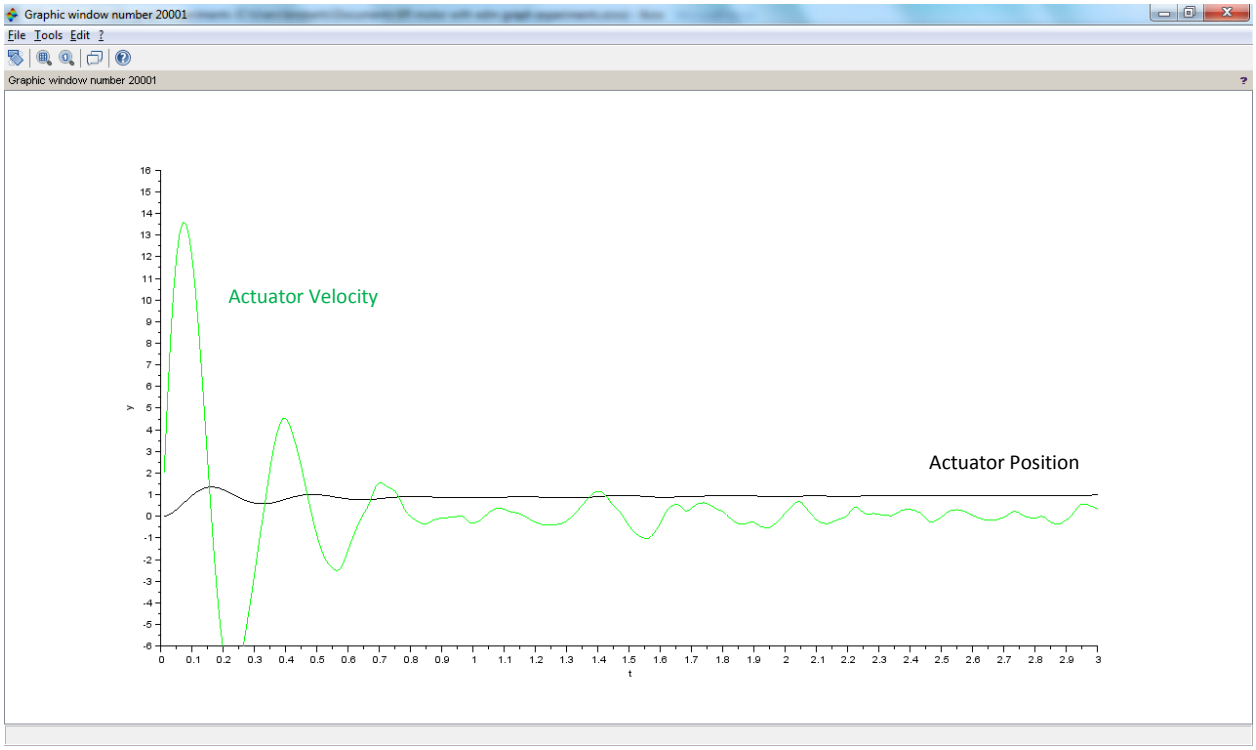


Figure 7.9 Scilab Simulation: Linear Actuator and EDM Process, Initial Three Seconds

7.1.2 Comparing Position Control and Velocity Control

The Scilab model in Figure 7.10 below shows a basic position control system, where a position error represents the EDM gap error, and random noise is applied to represent the EDM process. In this model the position error controls the motor armature voltage by means of a PID function (essentially armature voltage is proportional to error). Here, the Derivative coefficient of the PID controller is set to 0.001

The position (Black) and Velocity (Green) output of the simulation is as shown in Figure 7.11 below. Again, although the velocity of the actuator varies quite widely, the position remains fairly stable.

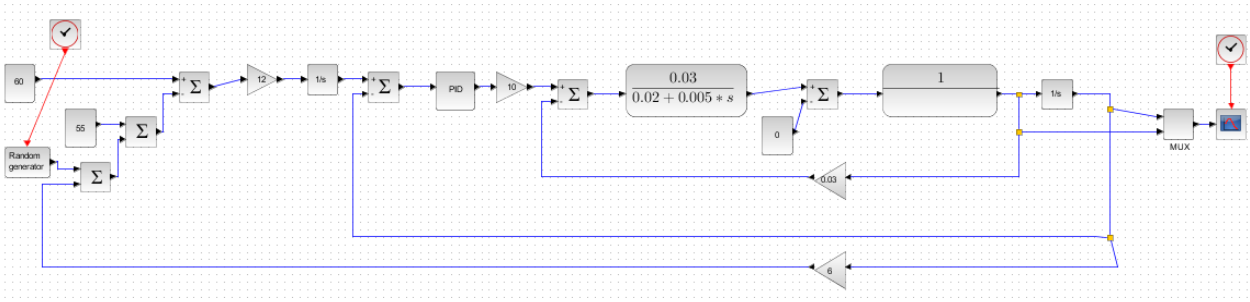


Figure 7.10 Scilab Model of Linear Actuator and EDM Process: Position Feedback

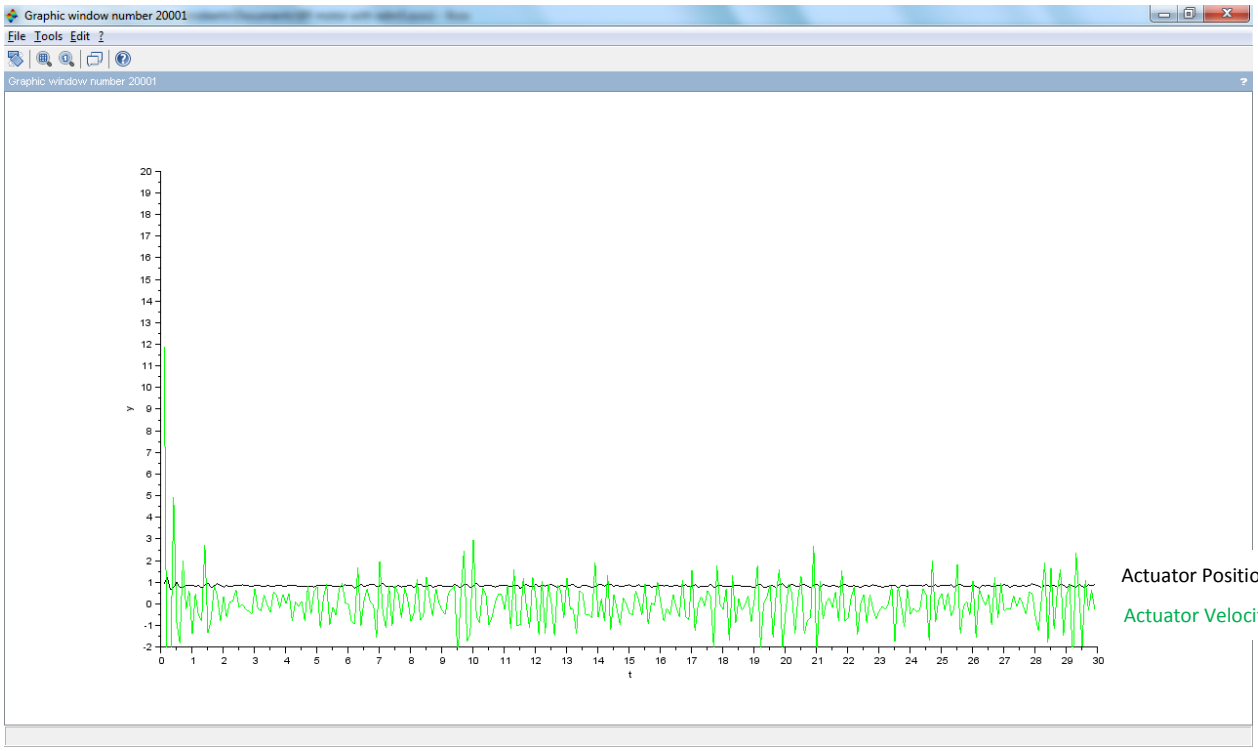


Figure 7.11 Scilab Simulation: Linear Actuator and EDM Process: Position Feedback

The process is seen to be stable, but when the derivative (D) co-efficient was set to 0, position was found to go out of control.

In contrast, when the control system is modified as in Figure 7.12 below such that the gap error produces a velocity command (instantaneous setpoint) to be given to the actuator (and another feedback loop forces obedience to this command, e.g. using an encoder to measure velocity (pulse frequency)), then control was seen to be stable even when the Derivative coefficient was reduced to zero. This illustrates the principle that a velocity feedback loop improves performance, which is in agreement with what was found practically with the LabVIEW program and the physical actuator. The output of simulation is shown in Figure 7.13 overleaf.

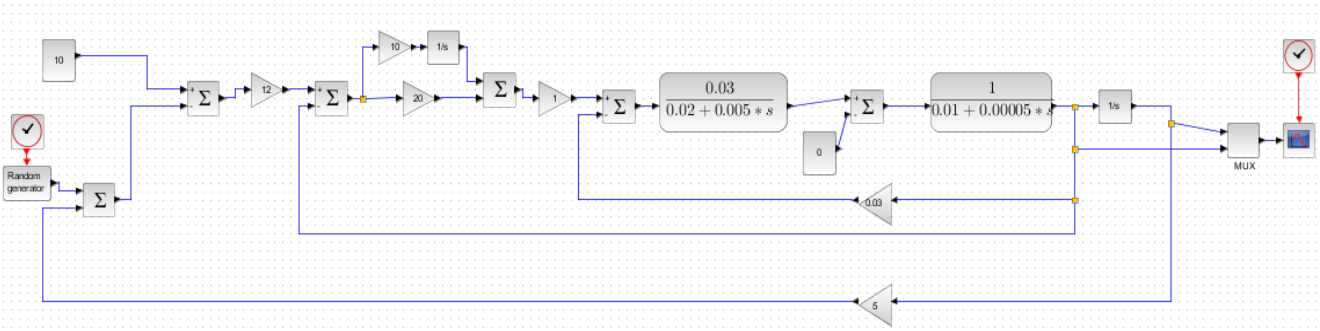


Figure 7.12 Scilab Model of Linear Actuator and EDM Process: Velocity Feedback

In the model, the outer feedback loop starts at Position (after the 1/s function after the actuator transfer function). Thus it may appear that this is a position feedback control system, but actually the outer loop is just representing the EDM process itself (but with no erosion over time effect), i.e. position is a surrogate for gap voltage condition.

It can be seen that there is velocity feedback, because the middle feedback loop starts at the right hand side of the actuator model, and the variable at that point is Velocity. (Innermost loop is the actuator itself, as previously discussed.) After Velocity comes the 1/s function, which is the integration function, getting Position from Velocity by integrating it over time. In this particular model, the PID block has been replaced with a separate P and I function (no D). The P is the 20 gain, and the I is the 10 gain (as the 10 gain has a 1/s function after it, meaning integration).

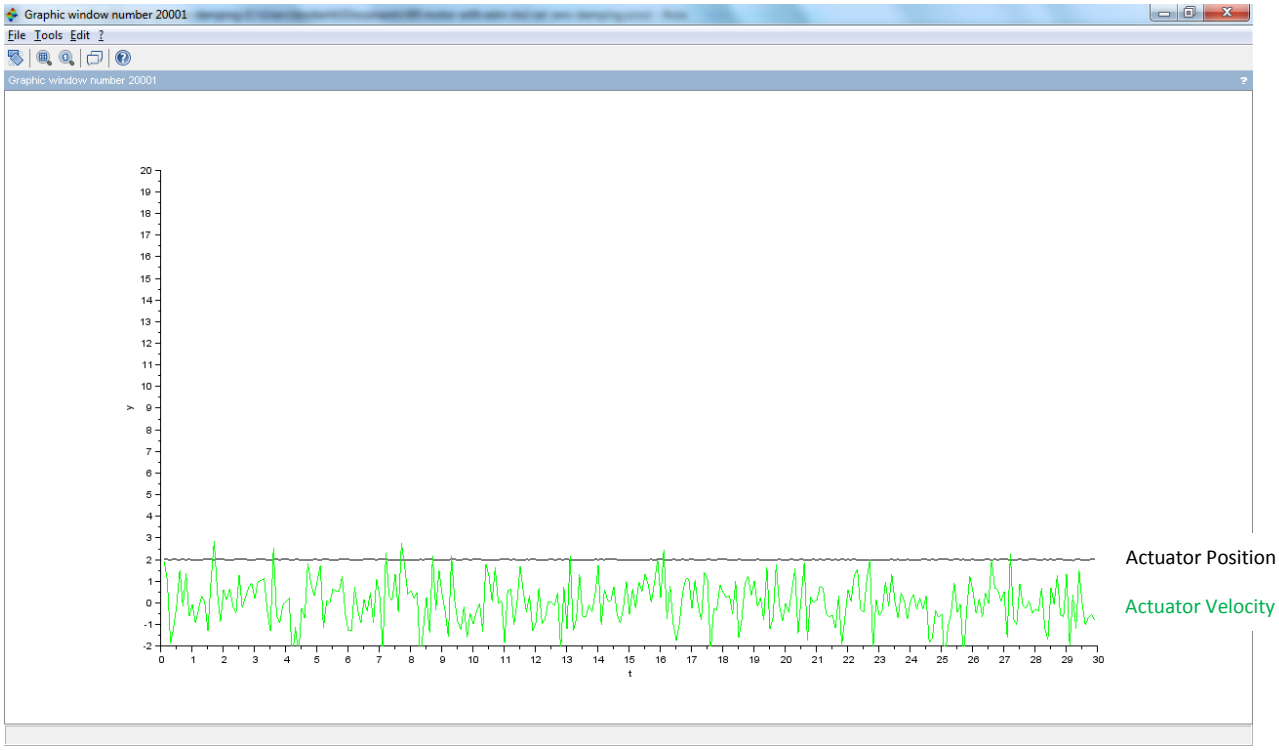


Figure 7.13 Scilab Simulation: Linear Actuator and EDM Process: Velocity Feedback

7.2 Determination of Linear Actuator Parameters for System Modeling

The actuator model could be improved by using parameters specific to the particular actuator. Generally the parameters can be quite easily measured. The following experiments were done in order to determine first the back-emf constant K_B, and second the force constant K_F.

To determine the back-emf constant, one simply needs to measure the voltage produced at the armature's terminals when the actuator is moved at a known velocity (and no current is flowing in the armature – i.e. open circuit essentially). No current must be flowing in the armature, since if there is, a voltage drop will occur in the armature due to V=IR, and the voltage measured at the armature's terminals will not be the full back-emf produced, but rather a reduced amount due to the resistance voltage drop. The actuator was moved manually; the voltage was measured in LabVIEW via the DAQ device, and the velocity was determined by considering the frequency of the encoder pulses received (i.e. finding known number of pulses in known time), also via the DAQ device. A program was developed to detect the peak velocity of movement and the corresponding voltage measured (i.e. occurring at the same instant). The program Block Diagram is shown in Figure 7.14 overleaf, and part of the Front Panel, showing results of a test, is shown on the page following (Figure 7.15).

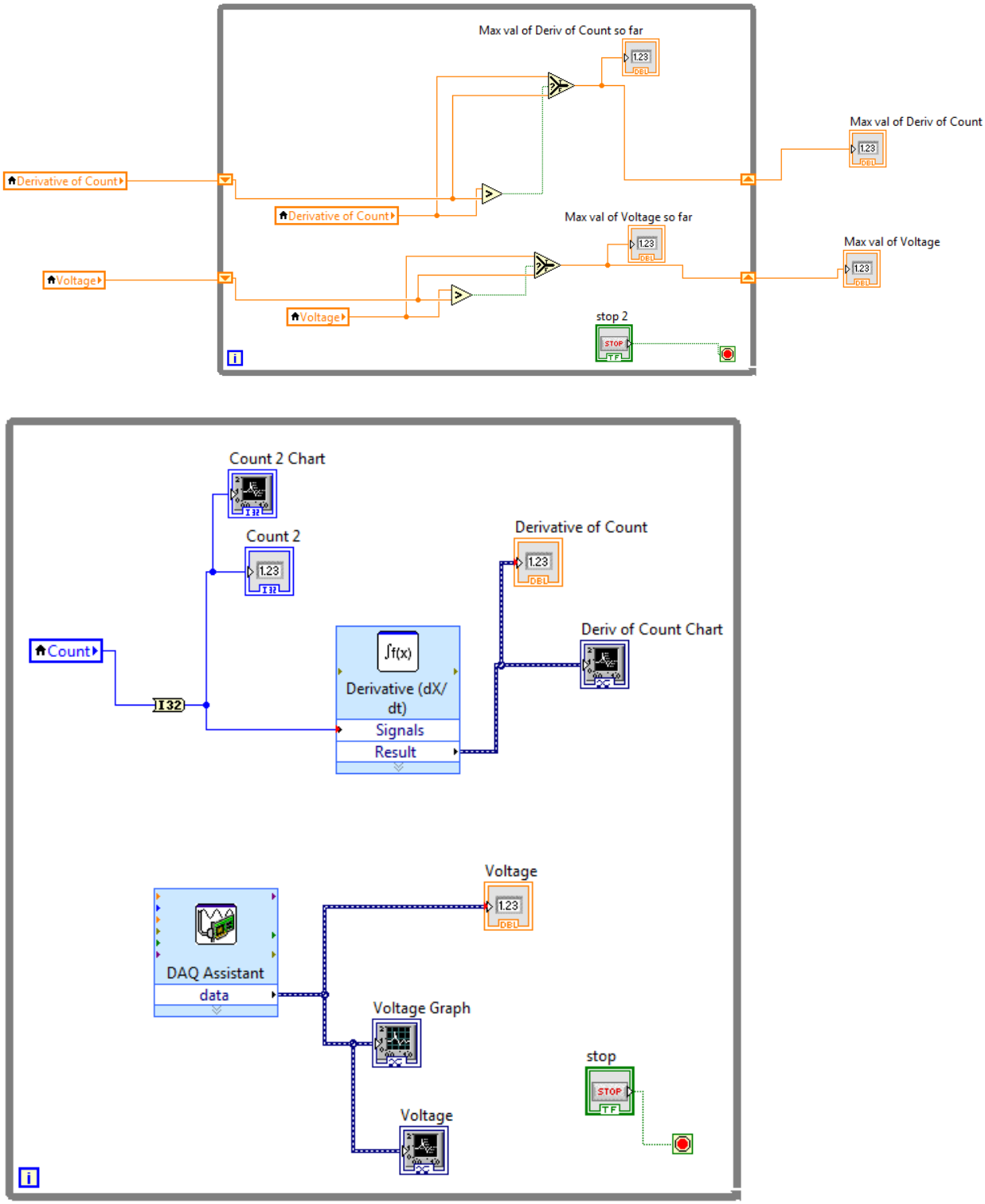


Figure 7.14 Actuator Back-emf Constant Test Program: Block Diagram

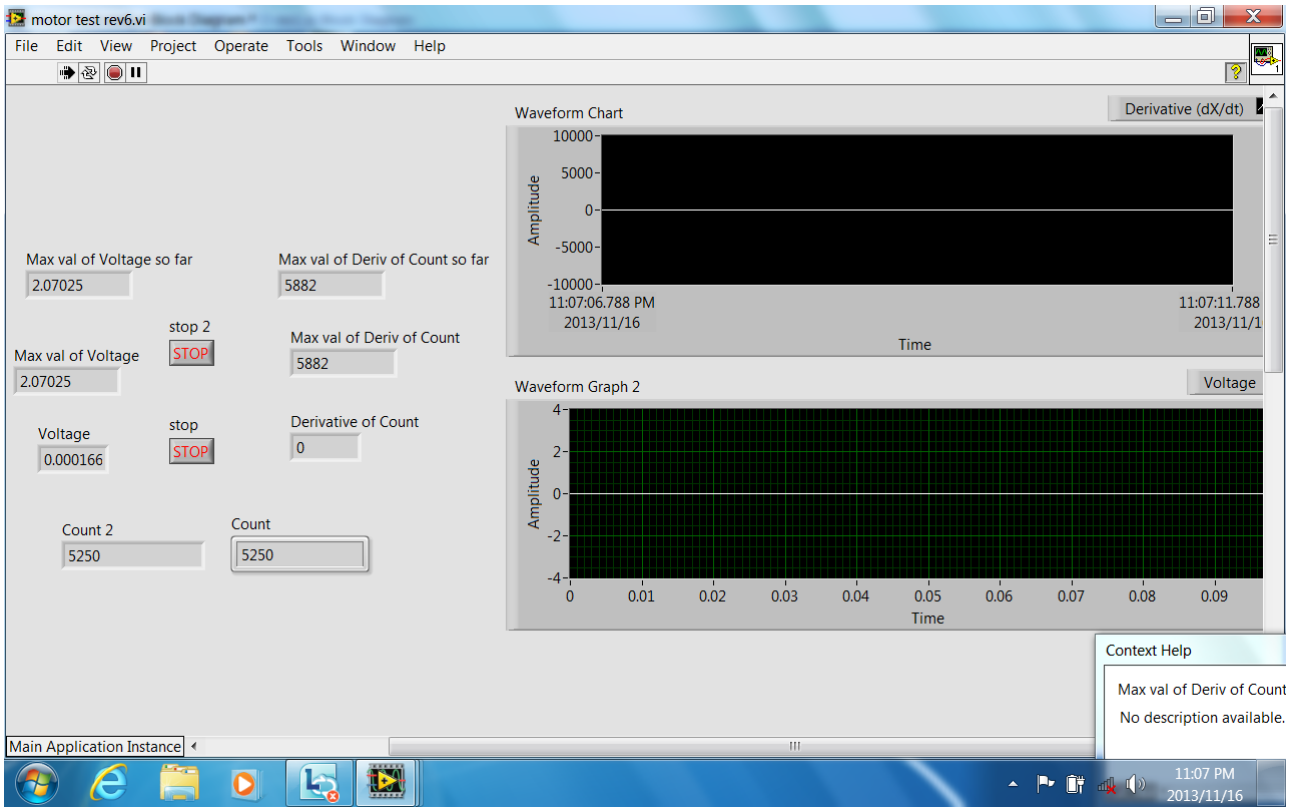


Figure 7.15 Actuator Back-emf Constant Test Program: Part of Front Panel

However, the results of this program were inconsistent and not entirely satisfactory, e.g. sometimes the maximum velocity captured 'thus far' was seen to change, yet no change was seen in the maximum voltage 'thus far'. The poor results were thought to be caused perhaps due to the maximum frequency being too fast for the DAQ device to capture (bearing in mind that the encoder resolution is 1 micron, so a very large number of pulses can be produced in a very short time if the actuator is sharply moved), or possibly the iteration rate of the LabVIEW 'Do' loop was not fast enough to detect all the (simultaneous) changes in velocity and voltage.

Instead, results seemed more consistent when the actuator was moved fairly slowly and steadily, and screenshots of graphs of frequency and voltage were taken. One such screenshot is shown in Figure 7.16 overleaf. Using the graph, one can find the slope of the Count (upper) graph and at the same instant see the magnitude of the voltage measured at the actuator's terminals. The slope of the Count graph gives the frequency of encoder pulses received. Since the divide-by-two function was being used, each pulse represents 2 microns. Thus if the frequency is multiplied by 2, velocity can be deduced (microns per second). For a motor, the back-emf constant is measured in, for

example, volts per rpm. For the actuator, volts per m/s is required. Thus if the voltage at any point in time is divided by the velocity at the same instant (as determined by the slope of the Count graph), the back-emf constant is obtained. Using the graphs, a value of approximately 8 volts per m/s was obtained (not a very accurate figure, as the measured voltage fluctuated quite a lot).

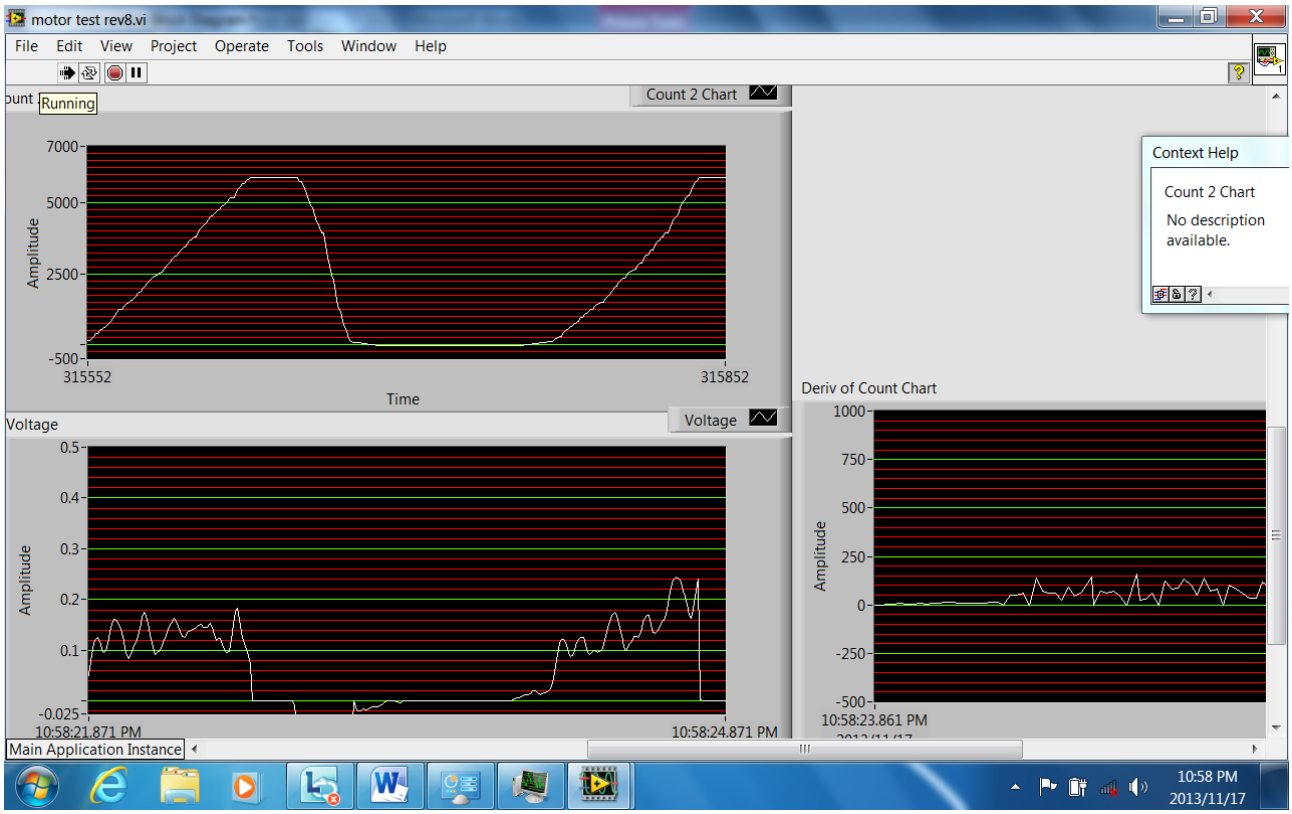


Figure 7.16 Actuator Back-emf Constant Test Results: Part of Front Panel

A simple experiment was also carried out to determine the force constant of the actuator. The results are shown in Figure 7.17 overleaf. To measure the force constant, one needs to measure the force produced when a known current is applied to the actuator, while there is no movement of the actuator. The current in turn can be determined by measuring the voltage applied to the armature, combined with a knowledge of the armature resistance, which can be measured directly using a multimeter. (There must be no movement of the actuator, since if there is movement, a back-emf is produced, which means that not all the voltage applied to the armature is seen by the armature's resistance, thus an error will be present.) The armature resistance was measured to be 12.4 Ω. The force produced by the armature coil was measured by mounting the actuator vertically, and resting known masses on it, producing known forces. Then the voltage (PWM) applied to the armature was slowly increased, until the actuator was seen

to just start rising. The current corresponding to this voltage is determined by I=V/R; the forces measured are then plotted against the currents, and the slope of the graph is the force constant. (Or any of the forces can be divided by the current required to produce it, and this will give the force constant.) When determining the forces to be used in the graph, the known mass of the moving part of the actuator was added to the weights of the masses, as one wants to determine the true force produced by the armature coil itself, not the net force available for accelerating an external load. The known masses were actually deduced from amounts of water added to a container having milliliter calibrations (and having a known mass), using 1 milliliter as equivalent to 1 gram, since the density of water is approximately 1000 kg/m^3.

The slope can be seen to be about 7 N/A, meaning that the force constant is about 7 N/A. In SI units, it can be shown that the back-emf voltage constant and the force constant should have the same value, and the results are roughly in agreement with this.

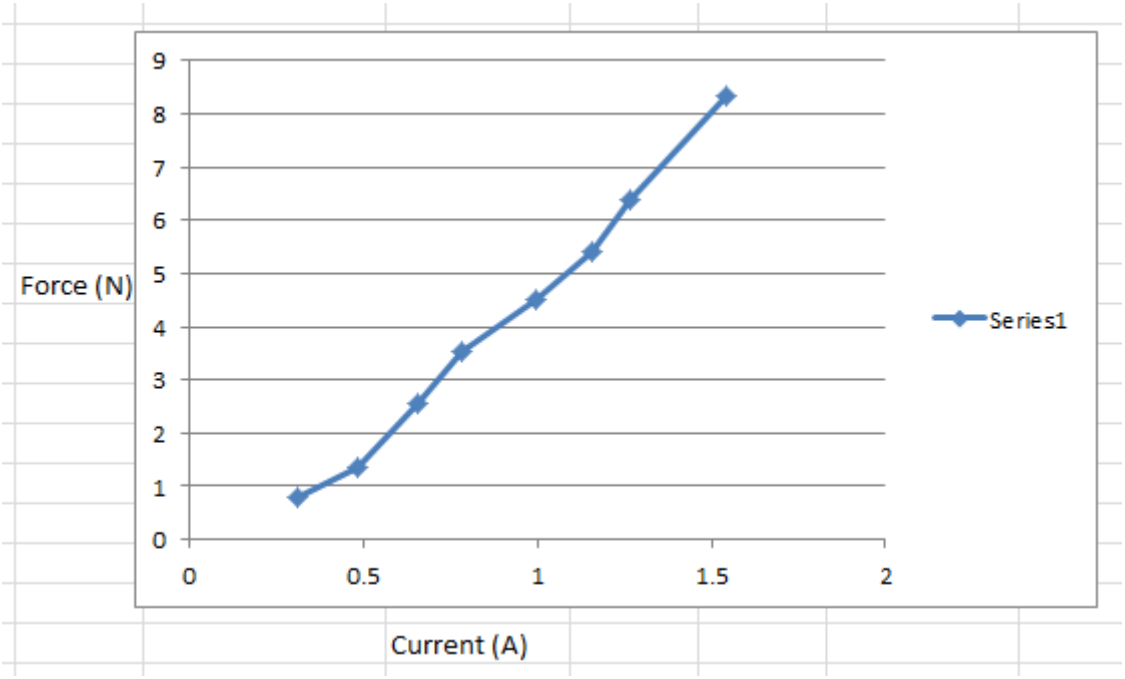


Figure 7.17 Actuator Force Constant Test Results

Inductance could be measured by determining (using e.g. an oscilloscope) the phase shift produced between applied voltage and current produced, when a known frequency sine wave voltage signal is applied to the armature, while it is held stationary. This test was not performed.

As an aside, an attempt to model the linear actuator in LabVIEW itself was also made, in the hope that the complete system could be simulated in LabVIEW (i.e. so that the program could run, with no physical actuator being required). A Laplace Transfer Function model of the actuator was successfully produced, but it could not be used as a 'function' in a main VI (the control program VI). Also the simulation speed for the model was very slow, taking quite few seconds just to show the actuator's response to e.g. a step or ramp command.

7.3 Conclusion to Chapter 7

This chapter shows various methods of modeling the controller, the actuator, and the EDM process itself, using the Scilab software program. The effect of adjusting certain parameters is noted. In addition, tests performed on the actuator to determine certain of its characteristic parameter values, were documented. The inclusion of a velocity feedback control loop was analyzed, and simulated performance was found to be more stable than in the case of a conventional PID controller, which is in agreement with what was found in practice in Chapter 6 above.

Chapter 8 following provides a discussion and conclusion to the research project as a whole. Also, areas for future research work are highlighted.

8. Discussion and Conclusions

Various results and levels of integration were achieved using the different approaches.

The EDM machine's servo drive was not capable of driving the CNC servo motors. A H-Bridge motor driver was able to be used in conjunction with the EDM machine's servo drive voltage, to drive one of the CNC machine's servo motors. EDM was achieved using this method, but control was poor, with a lot of electrode oscillation present.

A simplified control signal (i.e. the polarity) from the EDM machine was able to be read into the CNC's PC based Mach-3 program, and used to produce controlled advancement and retraction of the electrode, via the CNC tool's servo motors. Smooth, continuous EDM could unfortunately not be achieved using this method, although a certain amount of sparking did take place. It was noted that ideally access was needed to a variable in the interpolator of the PC based CNC program, if EDM was to be improved.

The research focused to a large extent on the use of an external controller which could be designed to suit, and would allow for the inclusion of a small linear actuator, enabling fine electrode motion, and the ability to produce co-ordinated multi-axis movement. The external controller was developed in LabVIEW using the ELVIS development board as a DAQ interface. This allowed filtering of the EDM gap signal to be achieved, as well as the implementation of PID control, the (potential) achievement of co-ordinated axis movements, and the production of the PWM signal required by the actuator driver. Although multi-axis movement was not actually produced, as only one actuator was acquired, the multi-axis LabVIEW programs catered in principle for additional axes.

Stable single axis EDM was achieved. Control loops that implemented velocity feedback inner control loops were found to be more stable than their counterparts. Stability was generally tested by manually applying a small but sudden disturbance (jolt) while the control program was running.

Stable EDM was achieved using a multi-axis program but with only one axis being physically used. The position error gives an important measure in a multi-axis environment and it was noticed that the position error was sometimes unacceptably high.

However, the integration function had been limited to prevent integral windup, and it is thought that when this restriction is lifted, the position error will reduce dramatically. In practice, probably only the version of multi-axis control employing a fixed iteration time for the position shift register (but allowing a varying value of increment/decrement) would be accurate, due to rounding errors because of the slow available update times of the shift registers. Probably a program employing mathematical integration instead of shift register based incremental integration would also prove more accurate.

An improvement that could be considered in future work would be to make the multi-axis programs able to accept multiple travel commands (including circular), whereas at present only one, linear, command can be set at time, to illustrate the principle of co-ordinated axis control. Also the current programs effectively employ 'incremental' mode of motion commanding, in that one can only tell the actuator where to project to, taking the present position as datum (whereas in 'absolute' mode, any position can be specified, regardless of present position, and the program will proceed to that position).

Certain glitches were found when using the LabVIEW control. In particular, there often seems to be a brief jump noticeable at the beginning of execution. This is possibly due to the initializing sequence of the variables in the program (a variable may jump initially to a certain value, before being told what value to assume).

Although the LabVIEW programs and linear actuator are at this stage independent of Mach-3 and the CNC machine's servo motors, an 'inch-worm' method has been described whereby Mach-3 and the CNC servo motors could work together with the linear actuator. This would be particularly useful for instances where long EDM travel is required. Also it should be noted that a LabVIEW program can be made to be an executable file, such that any PC can implement it, without having the LabVIEW software installed. Thus although significant development is involved, once achieved, it can be implemented cost effectively for the user. Also, the LabVIEW program could run on the same PC as the Mach-3 program, meaning a neater solution. A data acquisition device would however still need to be purchased by the user.

Another feature which could be added to the LabVIEW program is a periodic 'jump' function, often used in ED machines to assist with the removal of debris from the electrode-workpiece gap. This should not be difficult to implement, as a timer could be

used to override position commands (or rather to temporarily add a small distance to the current commands), such that the electrode briefly lifts and returns.

An interesting possibility for integration would be to let a conventional Mach-3 G-Code program, using any feedrate, ‘go through the motions’ in terms of producing pulses for its Gecko drives; these pulses could however instead be captured and recorded by a LabVIEW program, which could then ‘play-back’ the pulses, but regulated by the gap condition, i.e. the filtered gap voltage signal, such that motion gradually advances through the co-ordinated steps. This could be used for multiple actuators, or the pluses could even be re-routed to the CNC machine’s servo motors, via the Gecko drives, such that EDM could take place using the CNC machine’s existing motion producing capabilities. Control stability using the CNC servo motors would however not likely be as good as in the case of using linear actuators, due to the high inertia of the axes and the elasticity of the belt drives.

9. References

[1] Gorlach IA, Roberts BH, Simpson M, Estment W. **Reconfigurable Manufacturing System for Mould and Die Making.** PU-NMMU-NWU Consortium. 2009.

[2] Gorlach IA, Roberts BH, Simpson M. **Reconfigurable Manufacturing System for Mould and Die Making: Part Family Formation, Technologies and Process Planning Development.** Progress Report for: PU-NMMU-NWU Consortium. 2010.

[3] Landers R, Min B-K, Koren Y. **Reconfigurable Machine Tools**. CIRP Annals. 2001; 49(1): 269-274.

[4] Koren Y, Jovane F, Moriwaki T, Pritschow G, Ulsoy G, Van Brussel H. **Reconfigurable Manufacturing Systems.** CIRP Annals. 1999, 48:527-540.

[5] Xing B, Bright G, Tlale NS, Potgieter J. **Reconfigurable Manufacturing System for Agile Mass Customization Manufacturing.** 22nd International Conference on CAD/CAM, Robotics and Factories of the Future. 2006; 473-482.

[6] Katz R, Moon Y-M. **Virtual Arch Type Reconfigurable Machine Tool Design: Principles and Methodology.** NSF ERC/RMS, University of Michigan. 2000.

[7] Mehrabi MG, Ulsoy AG, Koren Y. **Reconfigurable Manufacturing Systems: Key to Future Manufacturing**. Journal of Intelligent Manufacturing. 2000; 11(4):403-419.

[8] Weller EJ, editor. **Nontraditional Machining Processes.** 2nd ed. Michigan: Society of Manufacturing Engineers. 1984..

[9] Ho KH, Newman ST. **State of the Art Electrical Discharge Machining.** International Journal of Machine Tools & Manufacture. 2003; 43:1287-1300.

[10] Groover MP. **Fundamentals of Modern Manufacturing: Materials, Processes, and Systems**. Wiley. 2007.

[11] Paul S. **Electro Discharge Machining.** Indian Institute of Technology, Kharagpur, Department of Mechanical Engineering. Manufacturing Processes II: Module 9: Non-Conventional Machining, Lesson 39 Ver. 2 ME.

[12] Yahya A, Manning CD. **Modelling, Simulation and Controller Design for Electro Discharge Machine System.** Electronic Systems and Control Division Research, Department of Electrical Engineering, Loughborough University, UK. 2003; 21-23.

[13] Chung C, Chao S-Y, Lu MF. **Modelling and Control of Die-sinking EDM.** WSEAS Transactions on Systems. 2009; 8(6):713-722.

[14] Behrens AW, Ginzel J, Bruhns F-L. **Threshold Technology and its Application for Gap Status Detection.** Journal of Materials Processing Technology. 2004; 149:310-315.

[15] Kunieda M, Lauwrens B, Rajurkar KH, Schumaker BM. **Advancing EDM through Fundamental Insight into the Process.** CIRP Annals – Manufacturing Technology. 2005.

[16] Chang Y-F. **VSS Controller Design for Gap Control of EDM.** JSME International Journal. 2002; Series C 45(3):712-721.

[17] Aristech, [Internet], Available from <http://www.aristech.com.tw/Frameset.htm> (Accessed 17 September 2010).

[18] Chang Y-F, Hong R-C. Parametric Curve Machining of a CNC Milling EDM. **International Journal of Machine Tools and Manufacture.** 2005; 45:941-948.

[19] Valentincic J, Brissaud D, Junkar M. **EDM Process Adaption System in Toolmaking Industry.** Journal of Materials Processing Technology. 2006; 172:291-298..

[20] Bayramoglu M, Duffill AW. **CNC EDM of Linear and Circular Contours using Plate Tools**. Journal of Materials Processing Technology. 2004; 148:196-203..

[21] Abbas NM, Solomon DG, Bahari F. **A Review of Current Research Trends in Electrical Discharge Machining.** International Journal of Machine Tools & Manufacture. 2007; 47:1214-1228.

[22] Mehrabi MG, Ulsoy AG, Koren Y. **Reconfigurable Manufacturing Systems: Key to Future Manufacturing.** Journal of Intelligent Manufacturing. 2000; 11(4):403-419.

[23] Sun X. **An Integrated Framework for Developing Generic Modular Reconfigurable Platforms for Micro Manufacturing and its Implementation.** PhD Thesis, Brunel University. 2009

[24] Altan T, Lilly B, Yen YC. **Manufacturing of Dies and Molds.** CIRP Annals – Manufacturing Technology. 2001; 50(2):404-422.

[25] Krajnik P; Kopac J. **Modern Machining of Dies and Mold Tools.** Journal of Materials Processing Technology. 2004; 157-158:543-552..

[26] DeVries MF, Duffie NA, Kruth JP, Dauw DF, Schumacher B. **Integration of EDM within a CIM Environment.** CIRP Annals – Manufacturing Technology. 1990; 39(2):665-672.

[27] Sure First, [Internet], Available from <http://www.surefirst.com.tw/html/02product_07.htm> (Accessed 6 December 2013).

[28] CNC Direct, [Internet], Available from <http://www.cncdirect.co.za/htm/mach.html> (Accessed 6 December 2013).

[29] CNC Direct, [Internet], Available from <http://www.cncdirect.co.za/htm/mach.html> (Accessed 6 December 2013).

[30] Geckodrive Motor Controls: G340 Installation Notes April 2008, [Internet], Available from <http://www.geckodrive.com/gecko/images/cms_files/G340%20REV-7%20Manual.pdf> (Accessed 7 January 2014).

[31] Applied Measurements Ltd, UK [Internet], LVDT & Position / Displacement Sensors [Internet], Available from <http://www.appmeas.co.uk/lvdt-position-and-displacement-sensors.html> (Accessed 6 December 2013).

[32] Pololu Robotics and Electronics, [Internet], Available from <http://www.pololu.com/category/94/pololu-simple-motor-controllers> (Accessed 7 January 2013).

[33] Fleming B. **Build a Pulse EDM Machine.** Author published. 2011.

[34] SMAC Moving Coil Actuators, [Internet], Product Flyer, LCA25 Series Linear Actuator, Available from <http://www.smac-mca.com/pdf/LCA25-010%20flyer%20(web).pdf> (Accessed 14 December 2013).

[35] ST Microelectronics, [Internet], L298 Dual Full-bridge Driver Datasheet, Available from <http://www.st.com/web/catalog/sense_power/FM142/CL851/SC1790/SS1555/PF63147> (Accessed 7 December 2013).

[36] Seeedstudio, [Internet], Available from <http://www.seeedstudio.com/depot/robotics-t-4.html?ref=top> (Accessed 14 December 2013).

[37] National Instruments, NI.com, [Internet], NI ELVIS, Available from <http://www.ni.com/ni-elvis/> (Accessed 14 December 2013).

[38] Kuo BC. **Automatic Control Systems.** Prentice Hall International Editions, 6th ed. 1991.

[39] **Motor Control.** University of Michigan: Module EECS461, Lecture 6, 2008 Available from: <http://web.eecs.umich.edu/~jfr/embeddedctrls/files/Lecture6.pdf> (Accessed 28 March 2014).

[40] Artsoft, [Internet], **Mach3-Mill: A User's Guide to Installation, Configuration and Operation**, Available from <http://www.machsupport.com/wp-ntent/uploads/2013/02/Mach3Mill_1.84.pdf> (Accessed 14 December 2013)

Appendix A: Summary of G-Codes

Table 1 Summary of G-Codes [40]

Summary of G-codes	
G0	Rapid positioning
G1	Linear interpolation
G2	Clockwise circular/helical interpolation
G3	Counterclockwise circular/Helical interpolation
G4	Dwell
G10	Coordinate system origin setting
G12	Clockwise circular pocket
G13	Counterclockwise circular pocket
G15/G16	Polar Coordinate moves in G0 and G1
G17	XY Plane select
G18	XZ plane select
G19	YZ plane select
G20/G21	Inch/Millimetre unit
G28	Return home
G28.1	Reference axes
G30	Return home
G31	Straight probe
G40	Cancel cutter radius compensation
G41/G42	Start cutter radius compensation left/right
G43	Apply tool length offset (plus)
G49	Cancel tool length offset
G50	Reset all scale factors to 1.0
G51	Set axis data input scale factors
G52	Temporary coordinate system offsets
G53	Move in absolute machine coordinate system
G54	Use fixture offset 1
G55	Use fixture offset 2
G56	Use fixture offset 3
G57	Use fixture offset 4
G58	Use fixture offset 5
G59	Use fixture offset 6 / use general fixture number
G61/G64	Exact stop/Constant Velocity mode
G68/G69	Rotate program coordinate system
G70/G71	Inch/Millimetre unit
G73	Canned cycle - peck drilling
G80	Cancel motion mode (including canned cycles)
G81	Canned cycle - drilling
G82	Canned cycle - drilling with dwell
G83	Canned cycle - peck drilling
G84	Canned cycle - right hand rigid tapping
G85/G86/G88/G89	Canned cycle - boring
G90	Absolute distance mode
G91	Incremental distance mode
G92	Offset coordinates and set parameters
G92.x	Cancel G92 etc.
G93	Inverse time feed mode
G94	Feed per minute mode
G95	Feed per rev mode
G98	Initial level return after canned cycles
G99	R-point level return after canned cycles

Appendix B: L298 Motor Driver

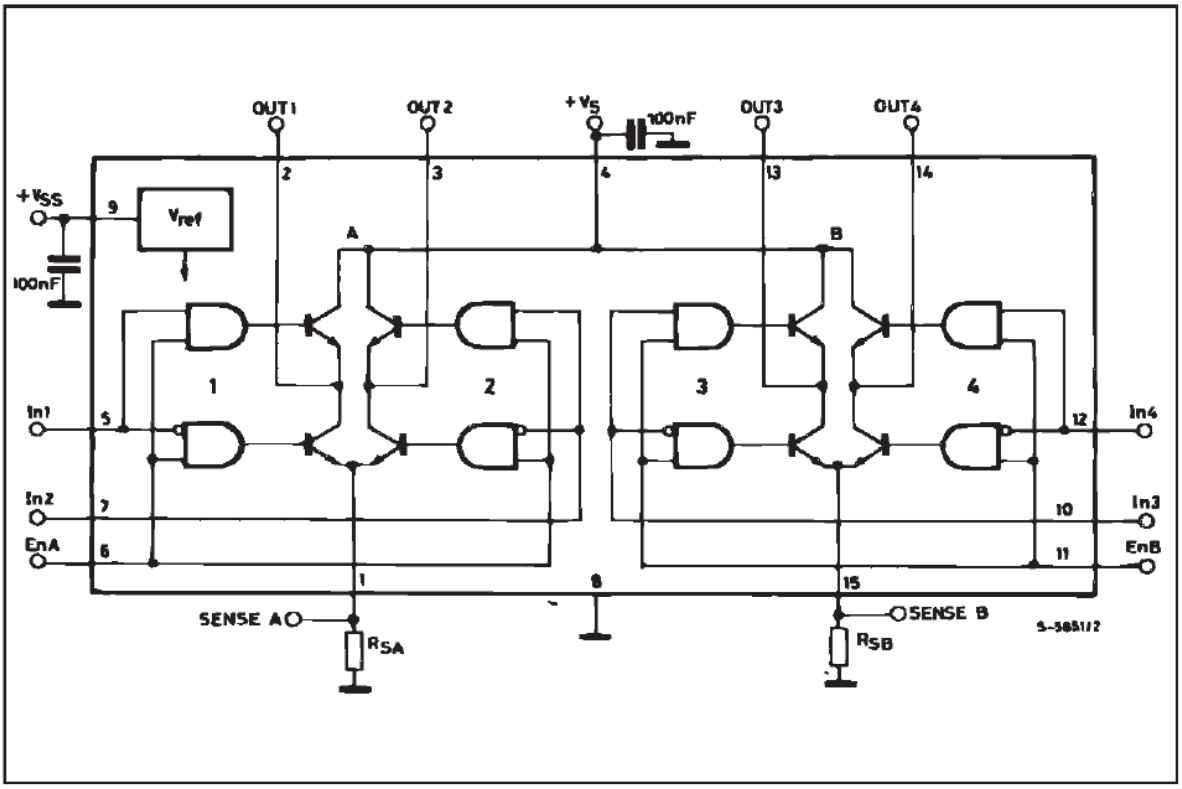
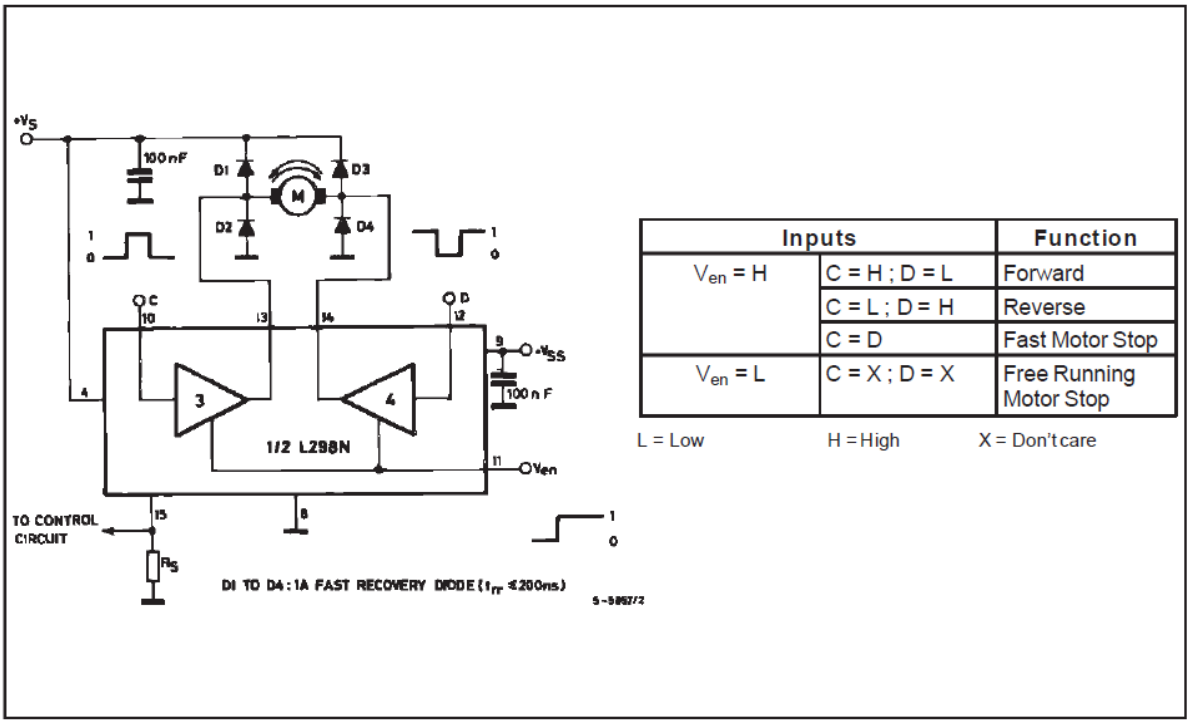


Figure B1 ST L98 Full-Bridge Motor Driver Schematic Diagram [35]



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figure B2 ST L298 Motor Driver, Bidirectional DC Motor Control [35]

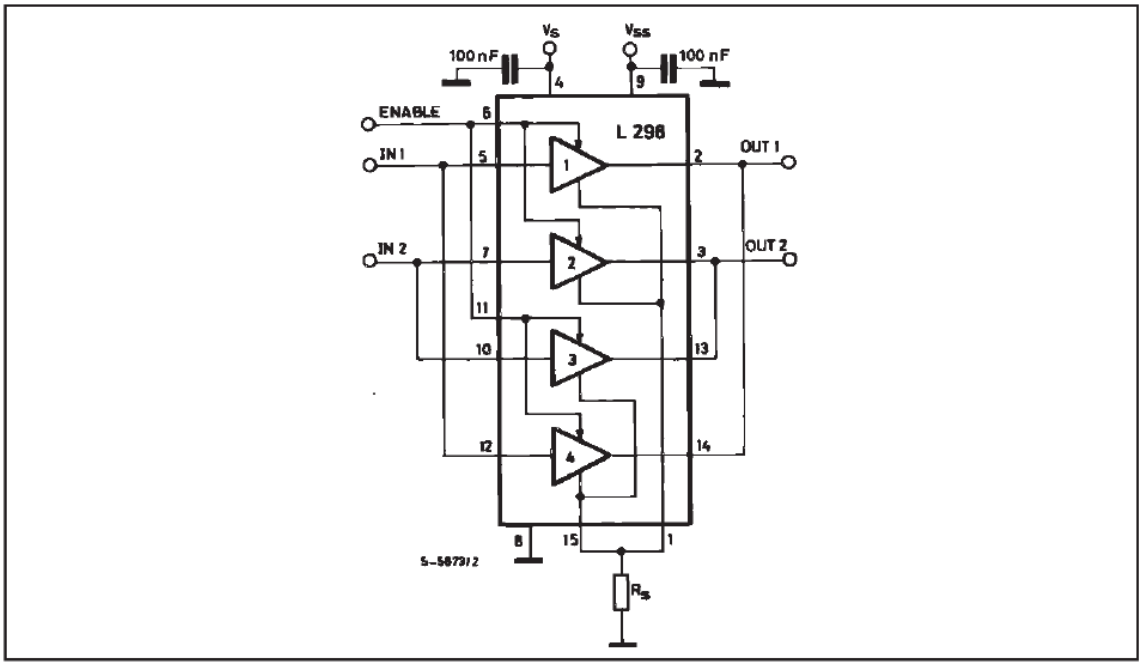


Figure B3 ST L298 Motor Driver, Outputs Connected in Parallel [35]