CONSTRUCTION AND CONTROL OF A DESKTOP EARTHQUAKE SIMULATOR

A Thesis Submitted to the
Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Civil Engineering
Structural Mechanics

by
Gökçe KINAY

July 2006
İZMİR

We approve the thesis of **Gökçe KINAY**

Date of Signature

……..
Assist. Prof. Dr. Gürsoy TURAN — **14.07.2006**
Supervisor
Department of Civil Engineering
İzmir Institute of Technology

……..
Assist. Prof. Dr. Serhan ÖZDEMİR — **14.07.2006**
Department of Mechanical Engineering
İzmir Institute of Technology

……..
Assist. Prof. Dr. Özgür Oğuz EĞİLMEZ — **14.07.2006**
Department of Civil Engineering
İzmir Institute of Technology

……..
Prof. Dr. Gökmen TAYFUR — **14.07.2006**
Head of Department of Civil Engineering
İzmir Institute of Technology

……..
Assoc. Prof. Dr. Semahat ÖZDEMİR
Head of the Graduate School

ACKNOWLEDGEMENTS

I would like to express my gratitudes and send my special thanks to my supervisor Asst. Prof. Dr. Gürsoy TURAN for his excellent supervision, help, guidance and encouragement he provided throughout my thesis. This thesis would not exist without his support.

I also would like to thank to members of the thesis committee, Asst. Prof. Dr. Serhan ÖZDEMİR, Asst. Prof. Dr. Özgür Oğuz EĞİLMEZ, Prof. Dr. Gökmen TAYFUR, and Asst. Prof. Dr. Faruk KEÇECİLER.

I also would like to thank to Asst. Prof. Dr. Thomas BECHTELER and to Asst. Prof. Dr. Zekeriya TÜFEKÇİ for their support in electronics and digital signal processing.

I would like to thank to Gökhan ERDOĞAN for all his support. He is a marvelous friend. I am very grateful to my friends Mr. Atalay Güven, Duygu Oğuz Kılıç, Evrim Yakut and Mine BAHÇECİ for their friendship, support, and encouragement.

Finally, I send my special thanks to my mother and father, İmren and Sinan for all their love and wonderful support for their little pampered daughter. I am grateful to my sister Tuğçe. Finally, it was wonderful to play with You when I could somehow manage to come home, Kutup, thanks a lot.

The shaking table project was made possible by research grants from IYTE, 2004-IYTE-19 and TUBİTAK project 'Desktop Earthquake Simulator' (Project no:MAG-HD-16(105M015)).

ABSTRACT

CONSTRUCTION AND CONTROL OF A DESKTOP EARTHQUAKE SIMULATOR

A portable, servo motor driven, and single-degree-of freedom earthquake simulator is manufactured. The moving table has a dimension of 40 cm x 40 cm. It can carry a load of 80 kg, accelerated to 2 g (1 g $\cong$ 9.81 m/s^2). Its maximum displacement capacity is ± 7 cm. In order to obtain the desired motion, a voltage of –10 to +10 volts is applied to the servo unit (motor driver), which is adjusted to move the table at –25 cm/s and 25 cm/s, respectively. A runtime program is written to read an earthquake's velocity-time data and to produce an electrical voltage that takes care of the following two items: First, the maximum speed of the simulated earthquake can not be larger than the motor's capacity. Second, the maximum earthquake displacement can not be larger than the table's displacement capacity.

In the present work, the recorded strong motion acceleration time series are processed in order to obtain useful data for engineering analysis. Strong motion accelerogram processing (earthquake data processing) is performed in Scilab. The objectives of strong motion data processing are corrections for the response of the strong motion instrument itself, and reduction of random noise in the recorded signals. The processing concentrates on the low-frequency ranges of the usable signal in the records.

The results obtained from comparison of the present work's outputs and some data providers' outputs are satisfactory. Some slight differences exist due to the different integration schemes and due to the application of different filter orders, zero-padding, and different filters for instance, acausal or causal Butterworth filter.

The simulations are performed in a regular Linux environment and also in a Realtime Linux environment. The advantage of the realtime environment ensures the signals send to the servo driver to be on-time – no delay due to operating system tasks.

ÖZET

BİR MASAÜSTÜ DEPREM SİMÜLATÖRÜNÜN İMALATI VE KONTROLÜ

Taşınabilir boyutta ve servo motor ile çalışan tek boyutlu bir deprem simülatörü imal edilmiştir. Hareketli tabla 40 cm x 40 cm'lik boyutlara sahiptir. 80 kg'lık bir yükü 2 g'lik ivmeyle (1 g $\cong$ 9.81 m/s^2) hareket ettirebilmektedir. Yer değiştirme kapasitesi ± 7 cm'dir. İstenen hareketi sağlamak için motoru –25 cm/s ile 25 cm/s aralığında süren motor sürücüsüne (servo ünitesi) –10 ile +10 Volt arasında bir değer verilir. Depremin hız-zaman datasını okuyup, elektrik voltajı üreten sürücü programı iki noktaya dikkat ederek hazırlanılmıştır: Birincisi, benzetilen depremin en büyük hızı motor kapasitesinden büyük olmamalıdır. İkinci olarak depremin en büyük yer değiştirmesi tablanın kapasitesinden büyük olmamalıdır.

Mevcut çalışmada, daha önce kaydedilmiş kuvvetli yer hareketi ivme zaman serileri, mühendislik analizlerine uygun veri elde etmek için işlenmiştir. Kuvvetli yer hareketi ivme serileri Scilab programında işlenmiştir. Kuvvetli yer hareketi verilerinin işlenmesinin ana amaçları, kuvvetli hareket kayıt cihazının kendi tepkisini düzeltmek ve kaydedilmiş sinyallerdeki rastgele gürültüleri azaltmaktır. İşlem, kayıttaki kullanılabilir sinyallerin düşük frekans alanlarına yoğunlaştırılmıştır.

Bu çalışmada elde edilen çıktılar ile bazı işlenmiş kuvvetli yer hareketi veri sağlayıcılarından elde edilen çıktıların kıyaslanmasıyla elde edilen sonuçlar memnuniyet vericidir. Bazı küçük farklar, farklı integrasyon yöntemlerinden, farklı filtre derecelerinden, sıfır ile beslemeden ve değişik filtre kullanımından (örneğin, acausal veya causal Buttherworth filter) kaynaklanabilir.

Simulasyonlar, Linux ve gerçek zamanlı Linux ortamında gerçekleştirilmiştir. Gerçek zamanlı Linux ortamının özellikleri sayesinde işletim sisteminin gecikmelere neden olmasına izin vermeden, sinyallerin servo sürücüsüne zamanında gönderilmesi sağlanmıştır.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

CHAPTER 1

INTRODUCTION

1.1. Overview

In this study a portable, servo motor driven, and one-dimensional earthquake simulator is manufactured. It is named as SHAKE1. The goal of this work is the reproduction/simulation of recorded earthquakes in a laboratory environment to test the response of model structures. Tests performed by shake tables are popular in areas of earthquake engineering, structural dynamics, structural control technologies (passive, active and semi active control devices), structural integrity testing of complete models, structural health monitoring, testing of smart materials, actuators, and sensors.

One of the interesting points of the current manufacture comes from the fact that servo motors are devices that are newly being used in shaking tables. They are clean devices when compared to hydraulic actuators, since there is no possibility of oil spillage. Also, the possibility of failure is expected to be less in a servo motor compared to a hydraulic actuator.

The largest size servo-driven shake table in the World is at the Earthquake Simulation Center, Ankara Civil Defence General Directorate. Its movable table has dimensions of 6.90 x 3.90 meters and its average payload is 6000 kgf with three degrees-of-freedom.

The earthquake simulator of NEES at Berkeley -The George E. Brown Jr. Network for Earthquake Engineering Simulation is the largest tri-axial table in the United States. It can shake structures weighing up to 45 tons by a maximum horizontal acceleration of 1.5 g.

One of the shake table manufacturer is Anco Engineers Inc. Over 40 ANCO tables are used by universities and industries world wide for simulation of earthquakes vibrations, shock, and automotive, rail and aircraft transportation. One of them belong to Koeri in İstanbul.

The earthquake simulator at National Center for Research on Earthquake Engineering, NCREE in Taiwan can simulate earthquake ground motions in six degrees-of-freedom. The shaking table is 5m × 5m and has a mass of 27 tons. Test specimen with a maximum payload of 50 tons can be accommodated on the table.

A few of the shake tables that are a part of the European Consortium of Earthquake Shaking Tables (ECOEST) are: 1) NTU Athens; 2) EERC Bristol; and 3) SMES Bergamo. These have six degrees-of-freedoms and a maximum specimen mass of 10, 15 and 30 tons, respectively. The one in Bristol can apply a maximum acceleration of 4,5*g (four and half times of the ground acceleration). Three degrees-of-freedom shake tables that are part of the ECOEST are the LNEC Lizbon and CEA Saclay. These have maximum specimen mass of 40 and 100 tons, respectively. The latter one has a maximum acceleration capacity of 4*g (four times of ground acceleration).

Earthquakes are recorded as accelerations or velocities (by digital instruments) in three components. These are the north-south, east-west and vertical directions. Two- , three- and six-degrees-of-freedom of shake tables are produced by some research groups and companies all around the world. One node has six-degrees-of-freedom in space, three of them are translational and the other three are rotational. Therefore a six-degrees- of-freedom shake table can perform the closest simulation to the real shaking of the ground during an earthquake.

SHAKE1 will be utilized in earthquake and structural dynamics lectures to visualize the dynamic response of any model structure and in presentations arranged with civilian associations to demonstrate earthquake effects. In addition, it is a starting step for a larger shaking table to be built at the Civil Engineering Laboratory, IYTE.

Last, for verification purposes, an accelerometer sensor is connected to the shaking table. It records the response of an earthquake simulation, which is then filtered and is compared to the input acceleration data. The filtering process is presented in Section 2.6.

1.2. The Aim of The Present Study

The primary purpose of this work is to manufacture a portable, servo motor driven, and one-dimensional earthquake simulator to simulate recorded earthquakes to test the response of some model structures against earthquakes. The secondary aim of the present work is processing of recorded strong motion acceleration time series in order to correct raw (recorded) earthquake data.

CHAPTER 2

SINGLE-DEGREE-OF-FREEDOM SHAKE TABLE

2.1. Introduction

In this study a single degree of freedom, desktop, servo motor driven shake table is manufactured. It is named as SHAKE1. Movement of the table is provided by the signals sent from the computer to the driver of the motor. Its primary function is to replicate the true nature of earthquake input to a test piece which is attached to the table in the laboratory.

Hardware parts consist of a single-degree-of-freedom shake table, a servo motor, a motor driver, a data acquisition card, its connectors, acceleration sensor, limit switches and interconnecting cables. The movement of shake table is triggered on an infinite screw by a servo motor.

Software consists of four different parts. The first one processes uncorrected earthquake acceleration data in Scilab. The second one is zero-position-calibration software. It provides that the motion of the table starts from the zero position. The third one sends analog output signals to the table and gets analog input signals from the acceleration sensor on the table. This code is written in C programming language for non real time applications. The last one is a Real Time Linux (RTL) C-code. It sends analog output signals to the table for real time applications. Studies continue on the RTL C-code to read analog input signals from the sensor for real time applications.

In order to obtain the desired motion, a voltage of –10 to +10 volts is applied to the servo unit (motor driver). The driver program reads an earthquake's velocity-time data and produces an electrical voltage. In addition, it takes care of the following two items: First, the simulated earthquake maximum speed can not be larger than the motor's capacity. Second, the maximum earthquake displacement can not be larger than the table's displacement capacity.

2.2. Specifications

2.2.1. Shake Table

SHAKE1 has a maximum load capacity of 80 kg and can shake this mass with a maximum acceleration of 2 g ($1g \cong 9.81\ m/s^2$). Its maximum displacement capacity is ± 7 cm. The moving table is made from aluminum with dimensions 40 cm x 40 cm x 1 cm. Additional specifications of the shake table are given in Table 2.1.

Table 2.1. Specifications of shake table

Specification	Value	Unit
Table overall dimensions (L x W x H)	60 x 50 x 9	cm
Table dimensions (L x Wx t)	40 x 40 x 1	cm^3
Total Mass	30	kg
Table Mass	4.5	kg
Maximum displacement	± 7	cm
Peak velocity	25	cm/s
Maximum force (theoretical)	1600	N
Maximum motor torque	1.274	$N \bullet m$
Servo motor power	400	W
Lead screw thread pitch	5	mm/rev
Ball nut dynamic loading capacity	780	$kg \bullet f$

The theoretical maximum force is defined in the following way. From the conservation of energy, the work done by the motor must be equal to the work done by the shaking table.

$$T \bullet \theta = F \bullet d$$

where, T is the motor torque, θ is the amount of rotation, d is the table displacement due to the rotation θ, F is the force to be calculated.

The lead screw thread pitch is 0.005 m/rev.

$$d = \frac{\theta}{2\pi} 0.005$$

Then the theoretical maximum force is calculated as follows:

$$F = \frac{2\pi \bullet 1.274}{0.005} \cong 1600N$$

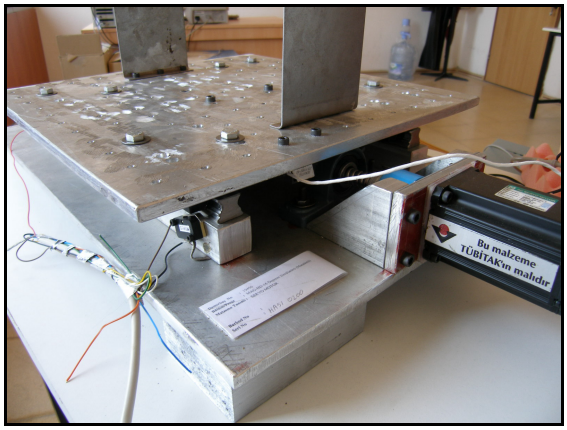


Figure 2.1. a) General view of SHAKE1

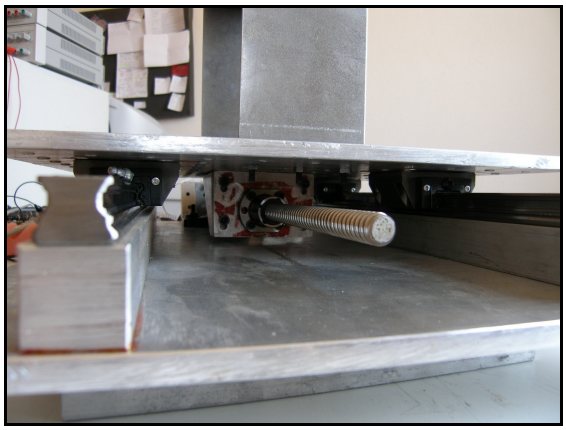


Figure 2.1. b) Side view of SHAKE1

The servo unit allows the motor speed to be adjusted to the corresponding maximum voltage. Maximum velocity of the table is related to the motor's speed and the lead screw thread pitch. The general view and view from one side of SHAKE1 is given in Fig.2.1. The servo motor drives the lead screw which lies in a ball nut that is mounted to the moving table (Fig.2.2). The table slides on frictionless rails which are mounted on motionless supports on two sides of the table. The desired motion is transmitted to the moving table (upper part of table) by means of servo motor – infinite screw – ball nut connection.

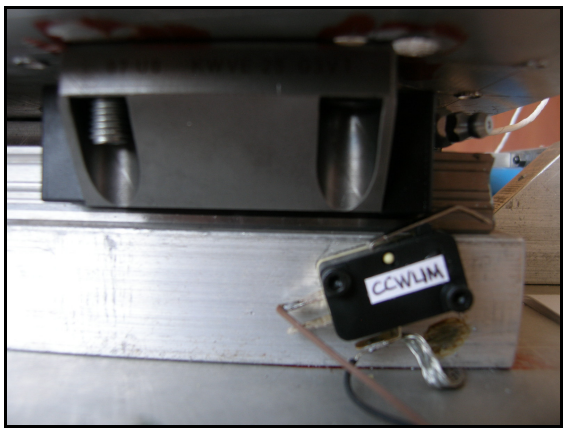


Figure 2.2. a) Limit switch

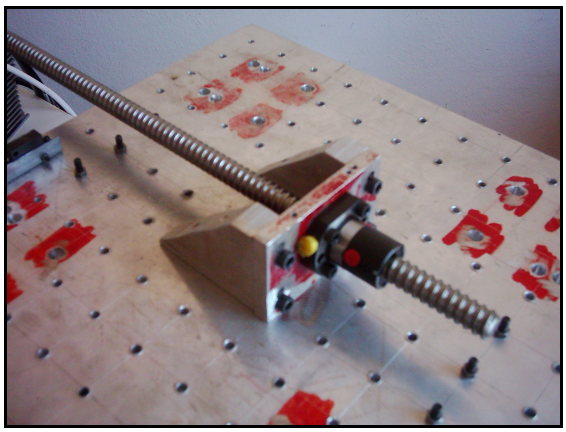


Figure 2.2. b) Ball nut of moving table

When the moving table reaches to the limits of the rail, an end switch is activated, which sends a signal to the motor driver. The motion is stopped in the related direction and the simulation continues in the reverse direction if the driving voltage changes its sign. This

should not happen, since the maximum displacement will be adjusted to the allowable amount by the runtime program.

2.2.2. Servo Motor and Driver

The schematic view and data flow of the Metronix AC servo motor and its driver are given in Fig.2.3. Servo motors are devices that are newly being used in shaking tables. Compared to hydraulic actuators, servo motors are less stronger, but much lighter in weight and more importantly, they are clean devices.

Velocity control is applied by the servo motor. The servo driver reduces the difference between the desired velocity and the table's real velocity. The feedback is performed by the encoder, connected to the motor shaft. An encoder is a device that converts rotary displacement into digital or pulse signals.

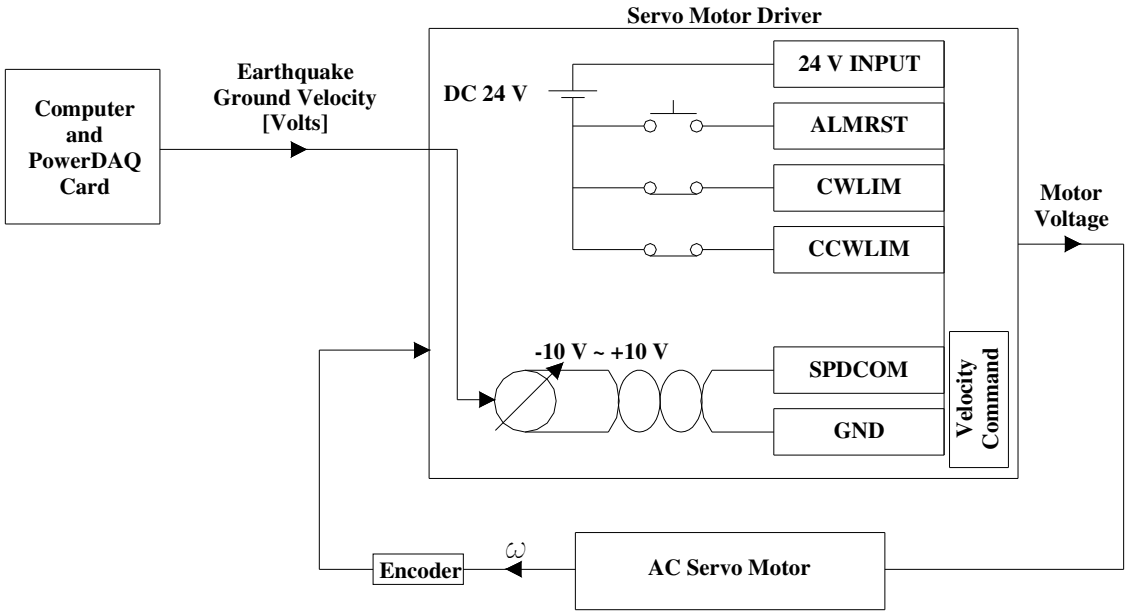


Figure 2.3. Schematic view of data flow

The earthquake velocity voltage is converted into rotational velocity by the servo driver and motor. The infinite lead screw converts it into linear velocity. As a conclusion, linear velocity is obtained by voltage sent to the servo driver as explained in Fig.2.4. Servo motor's rated rpm (revolutions per minute) is given as 3000 (50 rev/s) and this speed is achieved by a driving voltage of 10 Volts. Lead screw thread pitch is 0.5 cm. As a result of transformations, a velocity of 2.5 cm/s corresponds to 1Volt.

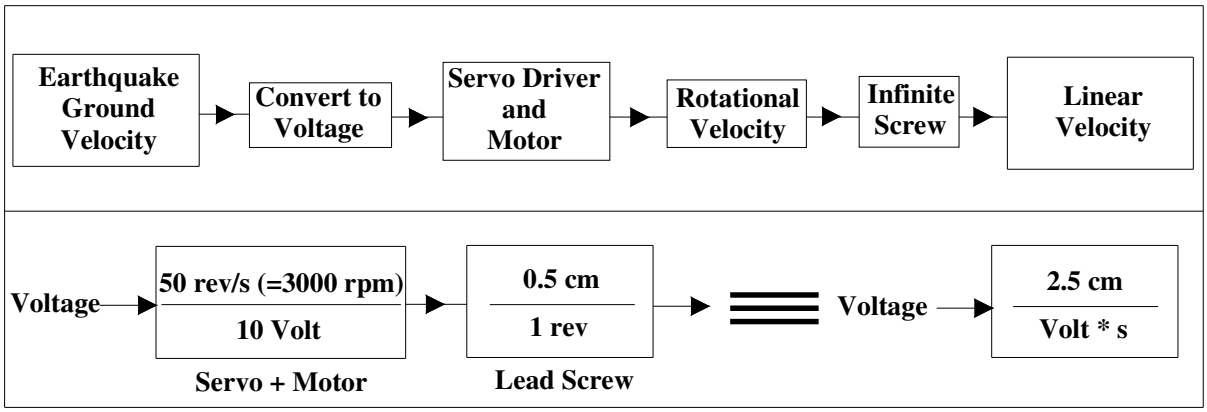


Figure 2.4. Motion transformation

2.2.3. Data Acquisition System

The computer interface is constructed with a PowerDAQ PDL-MF-16-50/16 (Multifunction lab board with 16 channels possesses a maximum speed of 50k samples /sec , 16-bit resolution) data acquisition card. It can read and send analog input (AI), analog output (AO), digital input/output (DIO) from single or multiple channels. PowerDAQ can work for non real time applications, as well as, for real time applications. It supports the Microsoft Windows XP / 2000 / NT 4.0 , QNX or Linux / RTLinux / RTAI Linux operating systems. It allows simultaneous operation of all subsystems (Analog In, Analog Out, Digital In, Digital Out and Counter/Timer).

In this project, a Real Time Linux 2.4.29-rtl-3.1 kernel is utilized. United Electronic Industries, which is producer of PowerDAQ, supports the following real time extensions of Linux: RTLinuxPro, RTAI, RTLinux GPL and Xenomai. The current real time Linux operating system is RTLinux GPL.

The outer connection parts of PowerDAQ can be seen in Appendix 24. Analog outputs can be sent from two channels by 12-bit resolution. Analog inputs are read from sixteen channels. System requirements for PowerDAQ board are PCI-bus system, a PXI-bus system or a CompactPCI-bus system with a free slot, Pentium-class processor, and a BIOS compliant with PCI Local Bus Specification Rev 2.1 or greater.

2.2.4. Acceleration Sensors

The table acceleration is measured by ADLXL202EB232 which is an evaluation board for the ADXL202 dual axis accelerometer with RS-232 interface and datalogging. It provides ability of acquisition of acceleration in two axis. It is a 2*g accelerometer can operate with a 275 Hz maximum sample rate (but this value is PC dependent).

2.3. Preparations

2.3.1. Manually Changing The Earthquake Table Position by Using The Servo Driver Unit

1. Switch on the servo driver.
   a) Connect the 24 Volt DC Power Supply to the Servo unit.
   b) Connect the 220 Volt AC Power Supply to the Servo unit.
   c) If an Error messages occurs in the servo panel, look-up its meaning in the Servo manual and fix the problem.
2. Go to PC-801 by up-down button on the servo driver unit.
3. Go to PC-803 by right-left button.
4. Press the Enter button that is located on the right-hand-side.
5. Apply speed by right-left button in the right-hand-side or left-hand-side direction.

(The servo driver unit can be seen in Fig.2.5.)

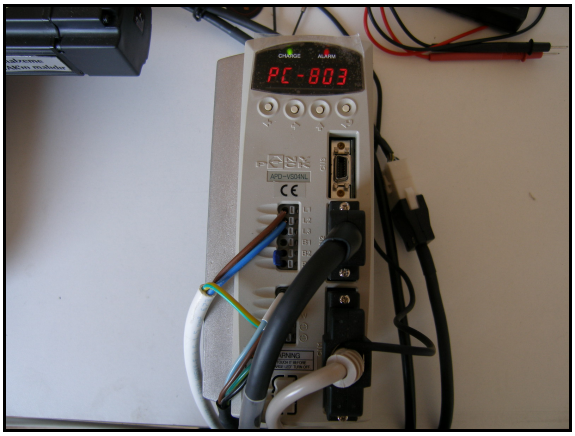


Figure 2.5. Servo driver unit

2.3.2. Zero-position Calibration Software

A calibration C code (ZeroPosiCalib.c) exists to bring the moving table to the middle (zero-position) of the bottom table and provides that motion starts from that position. An indiviual software for zero calibration was preferred instead of functions inserted in nonreal- and real-time applications.

At the beginning, a displacement of 20 cm is applied to the table. This is a distance that exceeds twice the table's maximum displacement. Displacements are applied as voltages as explained in Fig.2.4. Base of calculations is a velocity of 2.5 cm/s per unit Voltage. It is preferred to move the table slowly by appling 2 Volts for 4 seconds and waits until the 4 seconds elapses. The table reaches to one boundary.

Afterwards a displacement of 7 cm is given to the table in the reverse direction (Allowable limits of table are ± 7 cm.).

2.4. Non Real Time Application and Software

The PowerDAQ installation files come with example C codes that illustrate INPUT and OUTPUT of data. Among these, SingleAO.c and SingleAI.c are combined together and modified to send and read analog data in the same file. SingleAO.c performs a single update which is performed in a software timed fashion that is appropriate for slow speed generation (up to 500Hz). SingleAI.c performs a single scan acquisition which is performed in a software timed fashion that is appropriate for slow speed acquisition (up to 500Hz). The maximum data acquisition speed can be reached in case of buffered data. They are modified to send and scan multiple signals as appropriate for earthquake records. This code is given in A.18. It provides simulation of a recorded earthquake in a laboratory environment by sending earthquake velocity data to the servo unit and is designed to read acceleration data obtained from the sensor mounted on SHAKE1. Some explanations about SingleAIO.c can be found in A.19.

Time interval of signal sent to table should be equal to the time interval of recorded earthquake data. Reason of this situation is that it is desired to simulate frequency content of earthquakes as realistic as possible. The real effect of earthquake on a model structure can be seen, if the simulation is performed in real time. But, this is only possible while working with a real time operating system. At the beginning of the project a non real-time software was

constituted, namely SingleAIO.c given in A.18. Time arrangement was applied to obtain realistic time intervals for earthquake data. It is only composed of feeding the time interval after each time step a voltage that is equivalent to the corresponding EQ velocity is send to the servo drive and a loop, which does nothing, is run until the next time step is reached.

2.4.1. Scaling of Input Data

Earthquake ground velocity is fundamental input of the C code, therefore, it is given manually by *scanf* from the screen by redirection symbol < . Displacement and acceleration are read from text files by *fscanf*. Details about the format of input data files can be found in Section 2.4.4.

Absolute values of velocity data are calculated by macro *abs*. Maximum values of three sets of data are obtained by macro *max*. These are defined at the beginning of the C file. It is required to check if displacement data are in the allowable range of table ± 7 cm. In the following lines the flow of related part of the C code is given roughly.

$$DispScale = \begin{cases} 1, D_{\max} \leq 7\ cm \\ \dfrac{7}{D_{\max}}, D_{\max} > 7\ cm \end{cases}$$

where, $DispScale$ is displacement scale, $D_{\max}$ is maximum displacement.

If the limits of the shaking table are exceeded, scaling is applied to displacement, velocity, and acceleration data with the same scaling factor of *DispScale*. Same scaling factor for displacements, velocities, and accelerations is utilized due to the fact that time scale is accepted as $S_t = 1$.

$$S_V = \frac{S_L}{S_t}$$

$$S_a = \frac{S_L}{S_t^2}$$

where, S_L is length scale, S_V is velocity scale, S_a is acceleration scale, S_t is time scale.

As the second step, the maximum velocity is controlled if it is greater than 25 cm/s. If the limits are exceeded, scaling is applied to displacement, velocity, and acceleration data with the same scaling factor of *VeloScale*. In the following lines the flow of related part of the C code is given roughly.

$$VeloScale = \begin{cases} 1, V_{\max} \leq 25 \ cm/s \\ \dfrac{25}{V_{\max}}, V_{\max} > 25 \ cm/s \end{cases}$$

where, *VeloScale* is velocity scale, $V_{\max}$ is maximum velocity.

Related messages about unscaled (original) data and scaled ones, about scaling factors and maximum values are printed on the screen during execution of the driver software as given in Section 2.4.3.

The unscaled and scaled values of displacement, velocity and acceleration traces are given in Fig.2.6. The displacements, velocities, and accelerations of east-west component of Bolu station in 12 November 1999 Düzce earthquake are scaled twice. After the first scaling performed due to the check of maximum displacement, still some scaling is required. Because maximum scaled velocity is greater than the allowable value of 25 cm/s. The related scaling factors are given in Section 2.4.3.

Analog outputs of PowerDAQ have a fixed output range of +-10 Volts, therefore velocity data are converted into voltage values in range of $\pm$ 10 Volts in binary data form. For this conversion, maximum motor speed is 3000 rpm at 10 Volts and all velocity data are multiplied by a factor of *EqVoltWeight* which is equal to the division of maximum allowable voltage by maximum allowable table velocity.

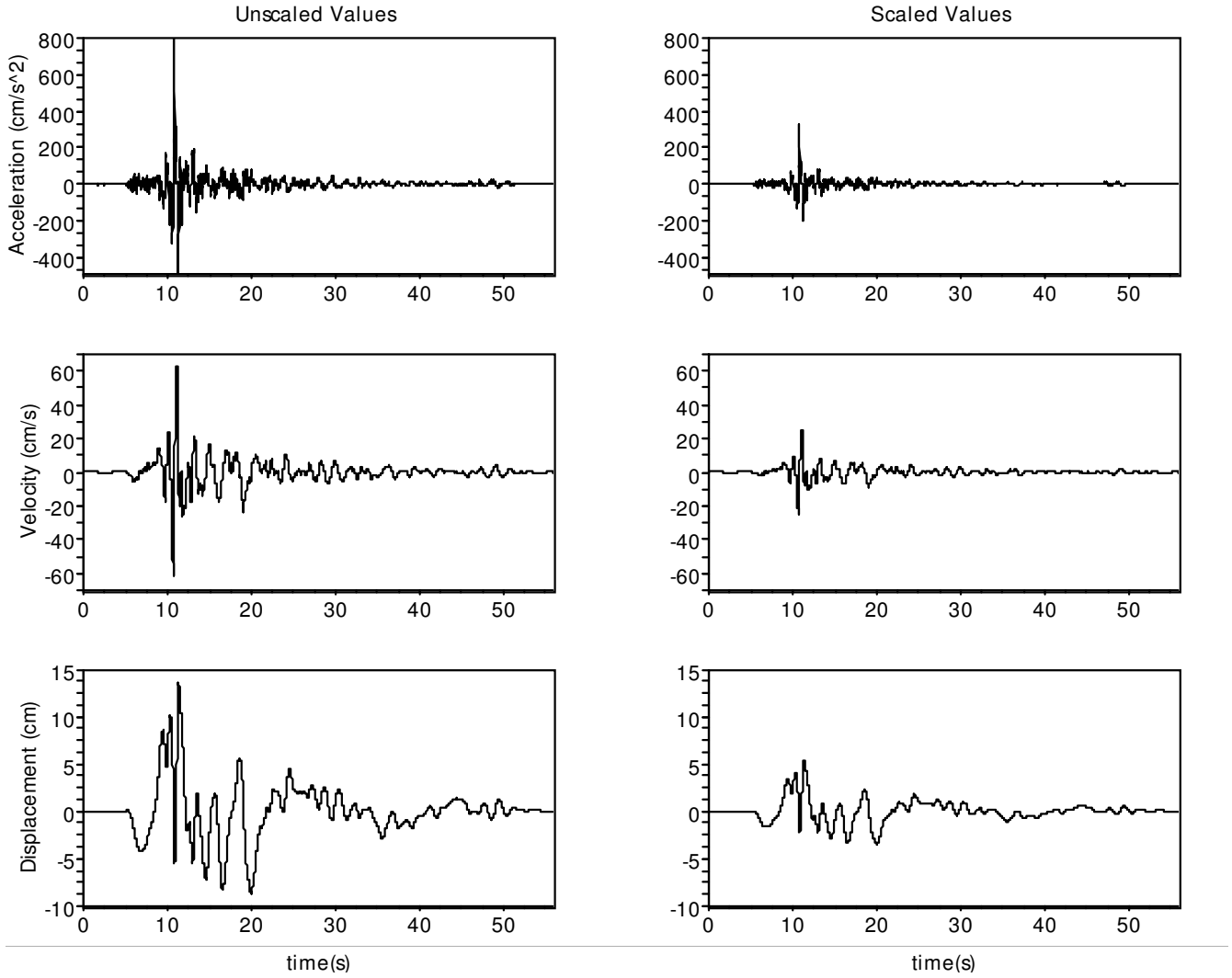


Figure 2.6. Unscaled and scaled values of 12 November 1999 Düzce earthquake 16:57 Mw=7.2 (Bolu station East-West component)

2.4.2. Running a Non Real Time Simulation with SHAKE1

1. Data files should only consist of numbers in form of a column vector and should not contain any alfa-numerical explanation.
2. Data files have been arranged in such a way that units of displacements, velocities, and accelerations are cm, cm/s and cm/s^2 (=gal), respectively.
3. Check the cable connections.
4. Switch on the servo driver.
5. Switch on the 24 V power supply.
6. >>./SHAKE1run < DataFile.txt

2.4.3. Messages Printed On The Screen During The Simulation

```
This is an PDL-MFx board

UNSCALED
max D : 13.548860 cm
max V : 62.069599 cm/s
max A : 806.802673 cm/s^2

ATTENTION
Table's allowable DISPLACEMENT limits are exceeded

FIRST SCALING
Scale factor :  0.516649
max D : 7.000000 cm
max V : 32.068176 cm/s
max A : 416.833496 cm/s^2

ATTENTION
Table's allowable VELOCITY limits are exceeded

SECOND SCALING
scale factor : 0.779589
max D : 5.457124 cm
max V : 25.000000 cm/s
max A : 324.958832 cm/s^2

CONVERSION INTO VOLTAGE
factor : 0.400000

index of max voltageAO = 1110
MAX VOLTAGE : 10.000000 Volt

End of data
```

2.4.4. Applying a New Earthquake Data

1. Velocity data of the same component of an earthquake are required. Time interval (dt) and number of data should also be known.
2. Be sure about the units of data. It should be in cm/sec unit.
3. Be sure that the velocities belong to the same component of record and to the same station.
4. Data files should only consist of numbers in form of a column vector and should not contain any alfa-numerical explanation.
5. Be sure that acceleration, velocity, and displacement data files are put in the current directory. Change acceleration and displacement data files names in C code. Velocity is fundamental input, therefore, is given manually from the screen by redirection symbol < .

>>./SHAKE1run < VelocityDataFile.txt

6. Change the scaled AVD data file names and the file name of data read by the accelerometer in the C code's *main* function by *FILE *name_of_link_to_the_file*.

2.5. Real Time Application and Software

A real time system is a system that can guarantee time requirements of the processes which are under its control. Similarly, a real time task is a task that has to be performed on time. Different than non real time applications, in real time systems it is important to perform a task within a specified time interval. It is desired to simulate earthquakes as realistic as possible. In this project's content, simulating an earthquake should be performed within the same time interval with the earthquake record. As the final step, a real time C code with RTLinux kernel is performed to send signals to the motor just on time.

2.5.1. A Brief Introduction to Real Time Linux

Linux is a standart time-sharing operating system and can be transformed into a RT-system by applying a patch from RT-Linux to the kernel's source code. Beside its real time property, RT-Linux is also a classical operating system. One can operate a real time task while performing any other activity on the computer.

A kernel module is simply an object file which contains routines and data to be loaded into a running kernel. It can be assumed that RT kernel is located between the standart linux kernel and the computer hardware as in Fig.2.7. Real time tasks possess the highest priority among all the tasks. RT-linux kernel gives the lowest priority to the standart Linux kernel. Real time tasks have direct access to the hardware, and they don't use virtual memory.

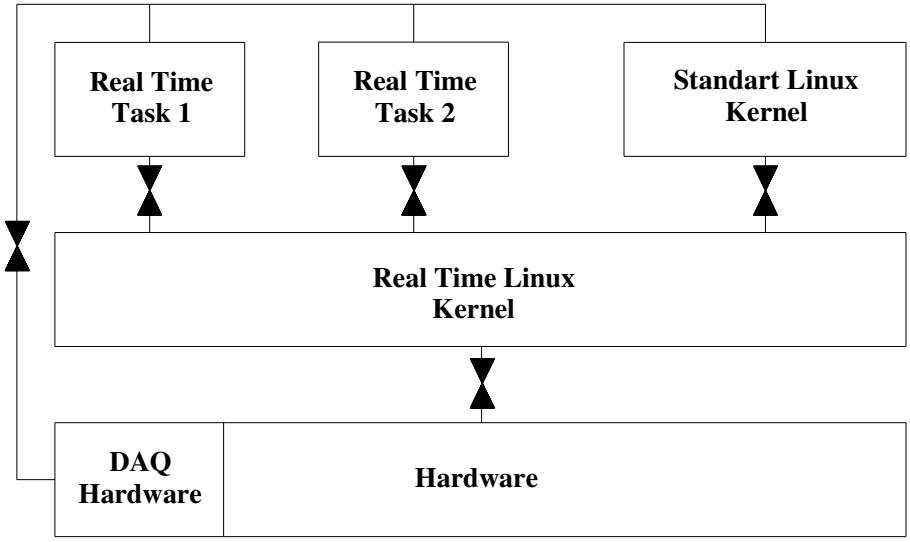


Figure 2.7. Real time Linux kernel

The functions *insmod* that loads a module into a running kernel, *lsmod* that shows modules in the running kernel, and *rmmod* that removes a module from a running kernel are mainly utilized for starting, observing, and stopping of a real-time process. One should be super-user (administrator) to utilize these functions.

Prior to the insertion of a module into the kernel, its dependencies should be loaded first. The modules that are used by the currently used analog interface card (PowerDAQ) use the modules rtl.o, rtl_time.o, rtl_posixio.o, rtl_fifo.o, rtl_sched.o are explained below:

rtl.o arranges core functionalities,
rtl_time.o controls processor clocks,
rtl_posixio.o provides a POSIX-like interface to device drivers,
rtl_fifo.o creates a real-time non-blocking FIFO implementation between real-time modules and user-space processes
rtl_sched.o implements a real-time scheduler.

Once a module is inserted to the real-time kernel, it becomes part of the operating system, hence it can use all the functions and can access all the variables and structures of the kernel. Unlike a regular executable program, a module that is to be inserted into a kernel can not have a *main* function. Instead, *init_module()* and *cleanup_module()* functions need to be present in the module. These are evaluated by the kernel when loaded and unloaded, respectively.

2.5.2. Running a Real Time Simulation with SHAKE1

1. Sign-in as super-user (root) from a terminal window (eg. konsole, or xterm, or rxvt).
2. Insert PowerDAQ by typing,

   >> modprobe pwrdaq
3. Insert the modules by typing,

   >> insmod /usr/src/rtlinux/modules/rtl.o

   >> insmod /usr/src/rtlinux/modules/rtl_time.o

   >> insmod /usr/src/rtlinux/modules/rtl_posixio.o

   >> insmod /usr/src/rtlinux/modules/rtl_fifo.o

   >> insmod /usr/src/rtlinux/modules/rtl_sched.o

   or, just by running a small *.sh file in which each of the above commands represent a line (This file can be found in A.19.)

   >> ./file-insmod.sh
4. Check if related modules have been inserted to the kernel by typing,

   >> lsmod
5. Go to the directory which contains relevant C file, *Makefile*, and data file.
6. Data file should only consist of numbers in form of a column vector and should not contain any alfa-numerical explanation. Unit of velocities are cm/sec.
7. Check *Makefile*'s relevant rows which contain C file's name.
8. Compile the C file by typing *make* in a konsole. From now on, object file *SHAKE1run.o* exists in the current directory. For kernels 2.4.x, object file exists with name of SHAKE1run.o and for kernels 2.6.x, object file exists with name of SHAKE1run.ko.
9. Check the cable connections.
10. Switch on servo driver.

11. Switch on the 24 V power supply.
12. Insert the object file to the kernel as a module

    >> insmod SHAKE1run.o

13. SHAKE1 simulates any earthquake applied to the system.
14. Print the messages of the C file on the screen

    >> dmesg

15. Remove module of object file to stop running after SHAKE1 has stopped running. Note that the file extension is not mentioned, unlike in step 13.

    >> rmmod SHAKE1run

16. If removing all the modules from kernel is required, type the following,

    >> rmmod pwrdaq

    >> rmmod rtl_fifo

    >> rmmod rtl_posixio

    >> rmmod rtl_sched

    >> rmmod rtl_time

    >> rmmod rtl

17. or run the file rmmod-file.sh in which each of the above commands represent a line (This file can be found in A.19.)

    >> ./rmmod-file.sh

2.6. Noise Study of Acccelerations Read by The Sensor

The east-west component of acceleration data of 12 November 1999 Düzce earthquake Bolu station is applied to the shaking table. The acceleration data are scaled before application to the table as explained in Section 2.4.1. The table's accelerations are measured during the simulation by means of a sensor mounted on the table. The signal coming from the sensor contains some amount of noise. Therefore, the raw signal's Fourier amplitude spectrum is evaluated for noise study and is given in Fig.2.8. The black line shows raw accelerations read by the sensor. By means of this figure, filtering techniques and filter corner frequencies are determined.

The highest frequency (Nyquist frequency) is equal to 50 Hz (= ½ dt). Therefore, a high-cut filter with a corner frequency higher than the Nyquist frequency will have no effect on the signal. Instead, the following applied strategy gave reasonable results for the two

derivatives of the acceleration signal. A low-pass Butterworth filter with a corner frequency of 40 Hz and a high-pass Butterworth filter with a corner frequency of 0.3 Hz are applied. Additionally, impulsive noises that can be seen in the time domain of the signal are removed.

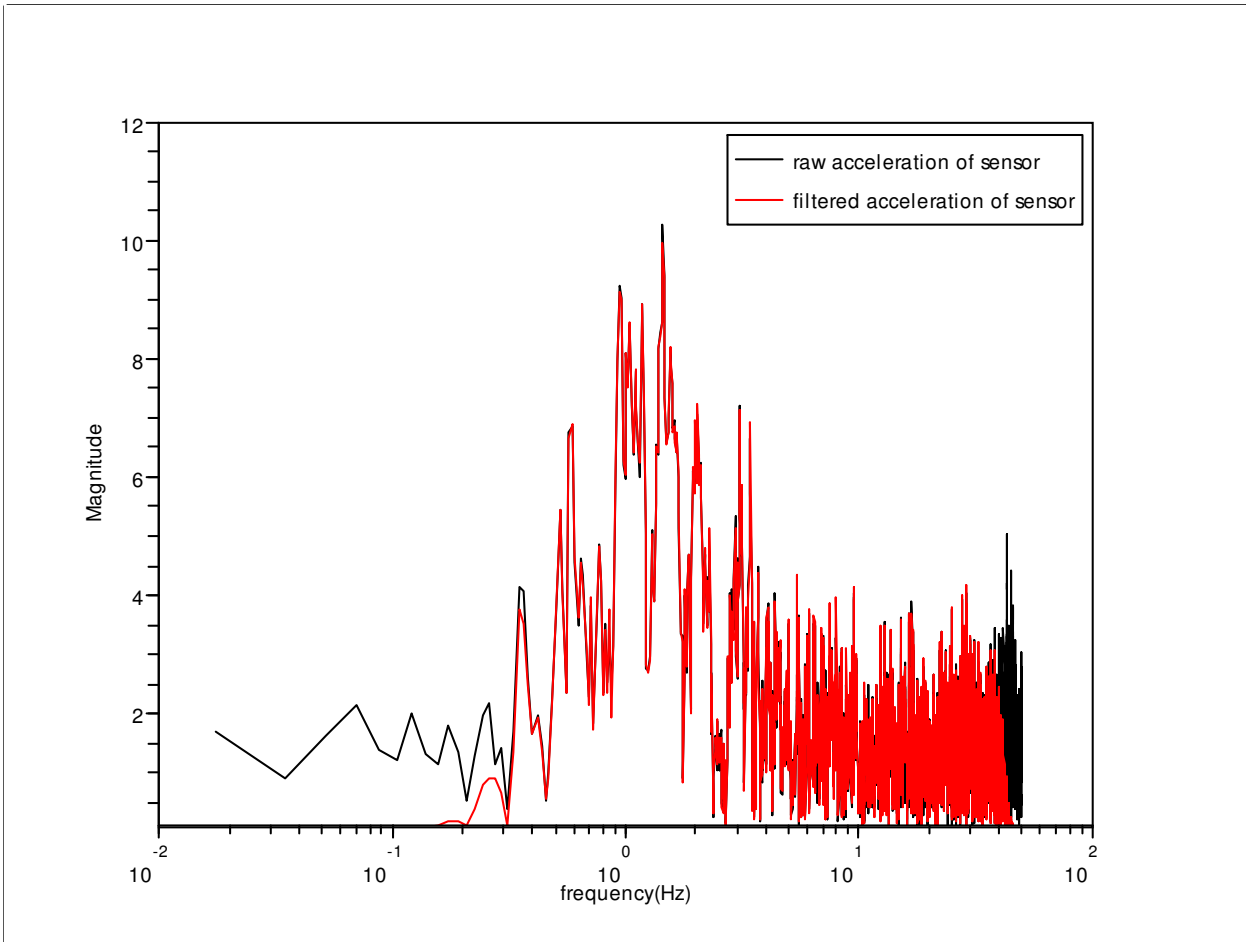


Figure 2.8. Fourier amplitude spectrum of raw and filtered accelerations measured by the sensor on the shaking table

Accelerations in time domain are given in Fig.2.9. The filtered accelerations, recorded by the sensor, and the applied earthquake accelerations are shown together in Fig.2.9. Accelerations that are measured and filtered are close to the earthquake acceleration record applied to the shaking table. Slight differences may come from the filtering techniques.

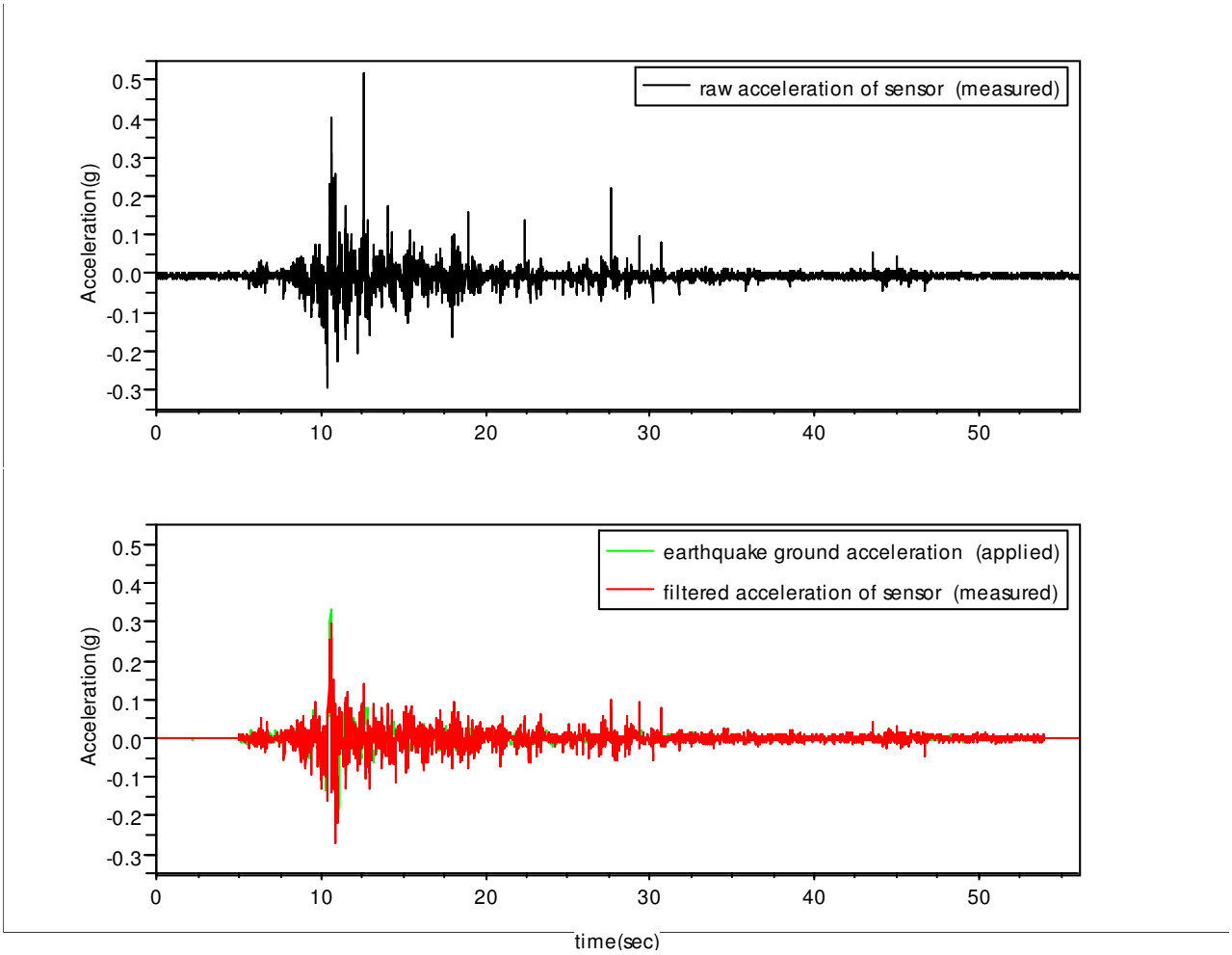


Figure 2.9. Applied earthquake ground accelerations and accelerations measured by the sensor on the shaking table (in time domain)

2.7. Warnings

1. dmesg contains all the messages since the previous restart of computer. In case of multiple runs of program, one can see previous messages on the screen.
2. Once any module is inserted to the RT kernel, it goes on running until the module is removed by typing,

   >> rmmod SHAKE1run
3. After a module is inserted and the real time application is finished, the module needs to be removed. Failing to do so, may cause unrealistic results when performing another real time task. This is particular true when changes are performed in the module file, saved under a different name, and inserted to the kernel. Check the

inserted modules by *lsmod,* and remove the relevant modules. This kind of error can be prevented by removing old modules

4. If a warning message of "*No powerdaq board detected*" occurs after run of program, this means that PowerDAQ is not inserted as a module to the kernel. Type the following as a super-user to solve this problem,

   >> modprobe pwrdaq

See if PowerDAQ is inserted to the kernel,

>> lsmod

If this solution does not work, compile PowerDAQ. For doing this, go to the powerdaq directory.

>> make clean

>> make (or >>make RTL=1 for free version of RT Linux)

>> make install

5. A warning message as given in the following lines may occur while inserting PowerDAQ. If a module is loaded that does not specify an approved license, the kernel is marked as tainted. You may safely skip this message.

```
karaburun powerdaq-3.6.6 # modprobe pwrdaq
Warning: loading /lib/modules/2.4.29-rtl-3.1/kernel/drivers/misc/pwrdaq.o will taint the
kernel: no license
 See http://www.tux.org/lkml/#export-tainted for information about tainted modules
Module pwrdaq loaded, with warnings
```

6. For each run,

   >> insmod SHAKE1run.o

and after the task has ended,

>> rmmod SHAKE1run

7. An error message of "*PdAcquireSubsystem failed*" occurs in case of a stop or interrupt to a running program by Ctrl+C or Ctrl+Z in the previous run. As a solution, exit from super-user Konqueror and then sign-in again as super-user.

2.8. Manufacturing Processes and Problems

Fundamental steps of this project are summarized as follows:

1. Processing of raw earthquake records.
2. Purchasing of materials, data acquisition card, servo motor unit.
3. Cutting and mounting of aluminum parts in university's metal shop.
4. Providing the communication among computer, data acquisition card, servo driver and motor. (Assembly of hardware)
    a. non real time software
    b. real time software

Problems during this project occurred at different steps. In order to obtain a perfect table movement, items were dismantled and mounted several times while making sensitive adjustments. In Fig. 2.10 a view from a mounting step can be seen.

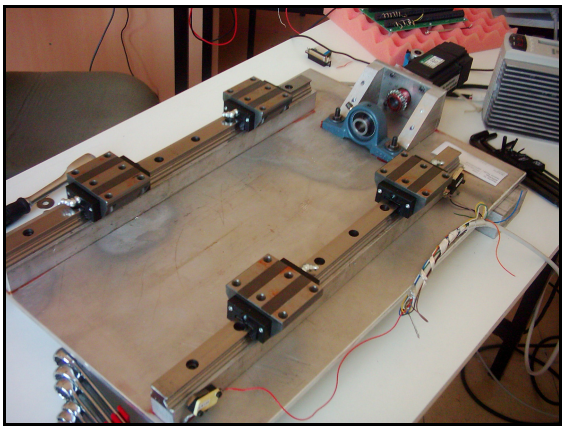


Figure 2.10. View from a mounting step (Shaking table without moving table and lead screw)

The main problem encountered during this study is the fact that the PowerDaq board does not have support for the COMEDI library and standards. Therefore, the board is limited to specific versions of the Linux kernel, RTLinux patches and the compiler. In case of purchasing a new data acquisition card one that is supported by the COMEDI library will be preferred. The COMEDI project develops open-source drivers, tools, and libraries for data acquisition. It is a collection of drivers for a variety of common data acquisition plug-in boards. The drivers are implemented as a core Linux kernel module providing common functionality and individual low-level driver modules.

CHAPTER 3

STRONG-MOTION ACCELEROGRAM PROCESSING

3.1. Introduction

Recordings from strong-motion accelerographs are of fundamental importance in earthquake engineering. The recorded strong-motion time series are called accelerograms. As it is usually the case, the records represent the acceleration of the recording instrument. Therefore, the related terms are named by using accel- prefix. But the time series may represent velocity rather than acceleration in case of digitally recorded data.

Earthquake strong ground motions are recorded by instruments that consist of three mutually perpendicular transducers (accelerometers), two of them measure the horizontal components of motion and the third one measures the vertical component of motion.

For engineering purposes, recorded ground acceleration time series during an earthquake is the most useful way of defining the shaking of the ground. Strong-motion accelerograms have provided earthquake engineers and seismologists valuable insight into the nature of earthquake ground shaking close to the earthquake source where damage can be expected. It is important to be able to estimate the level of unwanted noise signal present in each accelerogram.

Uncorrected records are those records which have not undergone any adjustment or correction. In extreme cases, which are out of interest of this work, uncorrected records may have been subjected to the removal of any obvious spurious peaks.

Strong motion data processing has two major objectives to make the data useful for engineering analysis. The first one is correction for the response of the strong motion instrument itself, and the second one is reduction of random noise in the recorded signals.

The main processing steps for acceleration time series are: baseline removal, conditioning and padding of the ends of the acceleration time series for subsequent filtering operations, instrument response correction and band pass filtering of the acceleration, integration of the corrected acceleration to velocity and displacement.

At the end of the ground shaking caused by an earthquake, the ground velocity must return to zero and this is indeed a criterion to judge the efficiency of the record processing.

The final displacement, however, need not be zero since the ground can undergo permanent deformation.

3.2. Analog Accelerographs

The first accelerographs were developed in the US in 1932. Analog (optical-mechanical) instruments were the first type of accelerograph developed and record the ground motion acceleration traces against time in the form of either a photographic trace on film or paper, or a scratch trace on waxed paper. The recordings, particularly those from analog instruments, invariably contain noise that can mask and distort the ground-motion signal at both high and low frequencies. The presence of this noise in the digitized time history and its influence on the parameters that are to be derived from the records should be identified. If the parameters of interest are affected by noise then appropriate processing needs to be applied to the records.

One of the disadvantages of analog accelerographs is that in order not to waste vast quantities of the recording medium, they operate on standby, meaning that they do not record all the time but are triggered by a minimum level of ground acceleration, usually of the order of 0.005 to 0.01g in the vertical direction. which means that the first motions are often not recorded. It is not possible to obtain the entire ground motion record.

The second disadvantages of analog accelerographs is related to their dynamic characteristics. The objective of the transducer in an accelerograph is that displacement response of a simple pendulum to be proportional to the acceleration of its base. The natural frequency of vibration of the pendulum must be much greater than the frequency of the motion being recorded. A pendulum of very high frequency would need to be extremely stiff, hence the displacement of its mass would be very small and to obtain a clearly decipherable record would require a large separation of the mass and the recording film, resulting in impractically large instruments. Therefore, the natural frequency of transducers in analog instruments is generally limited to about 25 Hz.

The most important disadvantage of analog instruments is that , after recovering the paper or film from the instrument, the trace of the strong ground motion is digitized. Otherwise, the recording can not be used in any engineering analysis. This is a time-consuming process and is one of the primary sources of noise.

The procedure for the noise in the digitized records is presented in the current part of the thesis. First of all, one should know that it is not possible to identify and remove all of the noise from the record and obtain the real seismic signal. The processing generally requires the removal of most of the record at frequencies where the Fourier amplitude spectrum shows a low signal-to-noise ratio. Portions of the record's frequency content where the signal-to-noise ratio is unacceptably low and portion of it that can be used with some confidence should be identified.

3.3. Digital Accelerographs

In the past twenty or thirty years, the digital instruments have been developed which record the strong ground motion in a digital form and provided solutions to problems related with analog accelerographs.

The separate digitization step is no longer required. They record on re-usable media (magnetic or solid state) and therefore can record continuously. The seconds just before the instrument triggered are called the pre-event time. By use of pre-event memory, they are able to retain the first wave arrivals, regardless of how weak they are. For this reason digital accelerographs record the entire ground motion that occurs during the earthquake. Their frequency range is much wider with transducers having natural frequencies of 50–100 Hz or even higher. In addition to all these advantages, direct digital recording provides more resolution than digitizing of an analog recording.

Although the digital instruments have become increasingly popular, there still exist more records from analog accelerographs than those from digital instruments in the databanks.

Digital accelerographs produce records much closer to the actual seismic signal than the analog ones. But some degree of record processing is still necessary as will be explained in the following sections.

3.4. Compatibility in Ground Motion Measures

"Data users are often troubled by the fact that if they integrate the acceleration time-history the velocities and displacements that they obtain do not match those provided by data distributors. In addition to this, the response spectra calculated from the acceleration time-

histories will often not match the response spectra provided by the distributor, at least, for example, in so much as the long-period displacements may not converge to the peak ground displacement." (Boore and Bommer)

In such cases, the data can be described as incompatible. If the velocity and displacement time-histories and the response spectra obtained from the accelerations match those provided, then it is compatible data. The causes of incompatible data are as follows:

1. The accelerations are filtered and then are integrated to velocities. Another filter is applied to velocity in order to reduce noise that is still present in the record. The process is then repeated on the displacements. The problems come from the fact that effects of the filters applied to the velocity and/or displacement are not carried back to the acceleration. Therefore the results from integration of the acceleration no longer match the velocity and the displacement that have been filtered.
2. The pads that are added for the application of the filter are removed and data is given to the user without pads. It is advised that by acausal filters, sufficient lengths of zero pads should be added to the records and these pads should not be removed from the filtered data. Otherwise, filter's effect will be spoiled resulting in offsets and trends in the baselines of the velocity and displacements obtained by integration. Additionally, removal of the pads affect the long period response spectral ordinates.

For the integration to obtain the velocity and displacement, zero initial conditions should be assumed. But, generally, one can come across with non-zero initial conditions in corrected velocity and displacement time series taken from the mentioned data providers on internet. The reason for this contradiction becomes clear by this explanation.

3.5. Noise Effect

Digitized accelerograms contain unreal 'motions' which do not belong to earthquake shaking. These are referred to as noise. The strong-motion data users should be aware of the fact that digitized accelerograms are never pure. The aim of processing accelerograms is the optimization of the balance between acceptable signal-to-noise ratios and the information required for a particular application.

The most important effects of noise in the record only become apparent when the acceleration trace is integrated to obtain the velocity and displacement time-histories Fig.4.5. The velocity and displacements obtained from integration of the accelerogram generally

appear unphysical. Such a ground displacement may be a single asymmetrical elastic displacement pulse of more than 10-100 meters amplitude. One of the reasons of unphysical nature of the velocities and displacements is the unknown boundary conditions. The initial velocity and displacement are both assumed to be zero. But, in reality, this is not the case. The signal prior to triggering could not be recorded in analog recorders, but it can be recorded by digital recorders.

The baseline problems are major matters encountered with both analog and digital accelerograms. Long-period noises generally come from the imperfections of digitizers or from systematic effects such as mechanical or electrical hysteresis in the transducer. Long-period noise can also be introduced by lateral movements of the film during recording and warping of the analog record prior to digitization. Recording seismograms can be seen in Appendix 15. The character of such offsets can range from apparent simple step-like offsets to complex, time-dependent variations, and multiple offsets.

Uncorrected records, especially analog ones, can be influenced by errors which will be most prominent in the high frequency ($\gtrsim$ 20Hz) and low frequency ($\lesssim$ 0.5Hz) ranges. Analog records are particularly affected by long period errors because of the digitization stage which is not required for records from digital instruments.

High frequency errors may have influence on estimates of the peak ground acceleration and short period spectral quantities. Low frequency errors influence long period spectral values and the velocity and displacement time-histories which are long-period quantities and are obtained by integrating the acceleration time history.

A model of the noise in the digitized record is required in order to estimate the signal-to-noise ratio. Most analog accelerographs produce traces which somehow symbolize noise effect and a number of studies have been done related with this subject. But these are out of interest area of this work.

An important advantage of digital recordings is that the pre- and post-event memory portions of the recordings provide a direct model for the noise in the record. But the most important component of the noise is actually associated with the signal itself. The pre-event memory does not contain the ‘signal-generated noise’ and therefore provides an incomplete model for the noise in the record.

3.6. Processing of Accelerograms

Generally the acceleration data obtained from analog or even digital strong motion instruments are subject to small baseline instabilities whick are step-like or transient offsets. At present, the reasons and characteristics of these offsets are not clear. In case of ground acceleration greater than 10 cm/sec^2 , some instabilities exist.

It is not possible to recognize baseline instabilities from acceleration records. By integrating accelerations to velocities and displacements, significant low-frequency signal distortions occur. The cause of these signal distortions are offsets of less than about 1 cm/sec^2 in recorded acceleration time series.

Some processing steps are applied in order to remove baseline offsets from recorded acceleration time series and to obtain reliable velocity, displacement, response spectra and Fourier amplitude spectra values. USGS-NSMP (U.S. Geological Survey National Strong Motion Program) , PEER (Pacific Earthquake Engineering Research Center) , KOERİ (Kandilli Observatory and Earthquake Research Institute) , ESD (the European Strong-Motion Database) , ANGORA-TKYHP (General Directorate of Disaster Affairs - Earthquake Research Department-Turkey National Strong Motion Program) are some of the unprocessed and processed data providers.

The important processing steps applied by PEER are given as a flowchart in Fig.3.1. The processing of the strong motion records in the PEER database is in general different than the processing done by the agency that collected the data. The differences should be small within the frequency passband common to both processing procedures.

The processing starts with the strong motion data from data provider. In case of analog records a digitizing step is necessary, which introduces considerable noise over a wide frequency range.

The instrument response is transformed in the Fourier domain for the instrument's amplitude and phase. High- and low-pass causal Butterworth filters are applied at frequencies, which the earthquake ground motion in the recorded signal significantly exceeds the noise level. The filter frequencies are determined for each component of a record by means of the Fourier amplitude spectrum and the integrated displacement time history *D(t)*.

Beside the procedure applied in the PEER, a combination of different processing procedures applied by USGS and KOERİ is given in Fig.3.2 as flowchart.

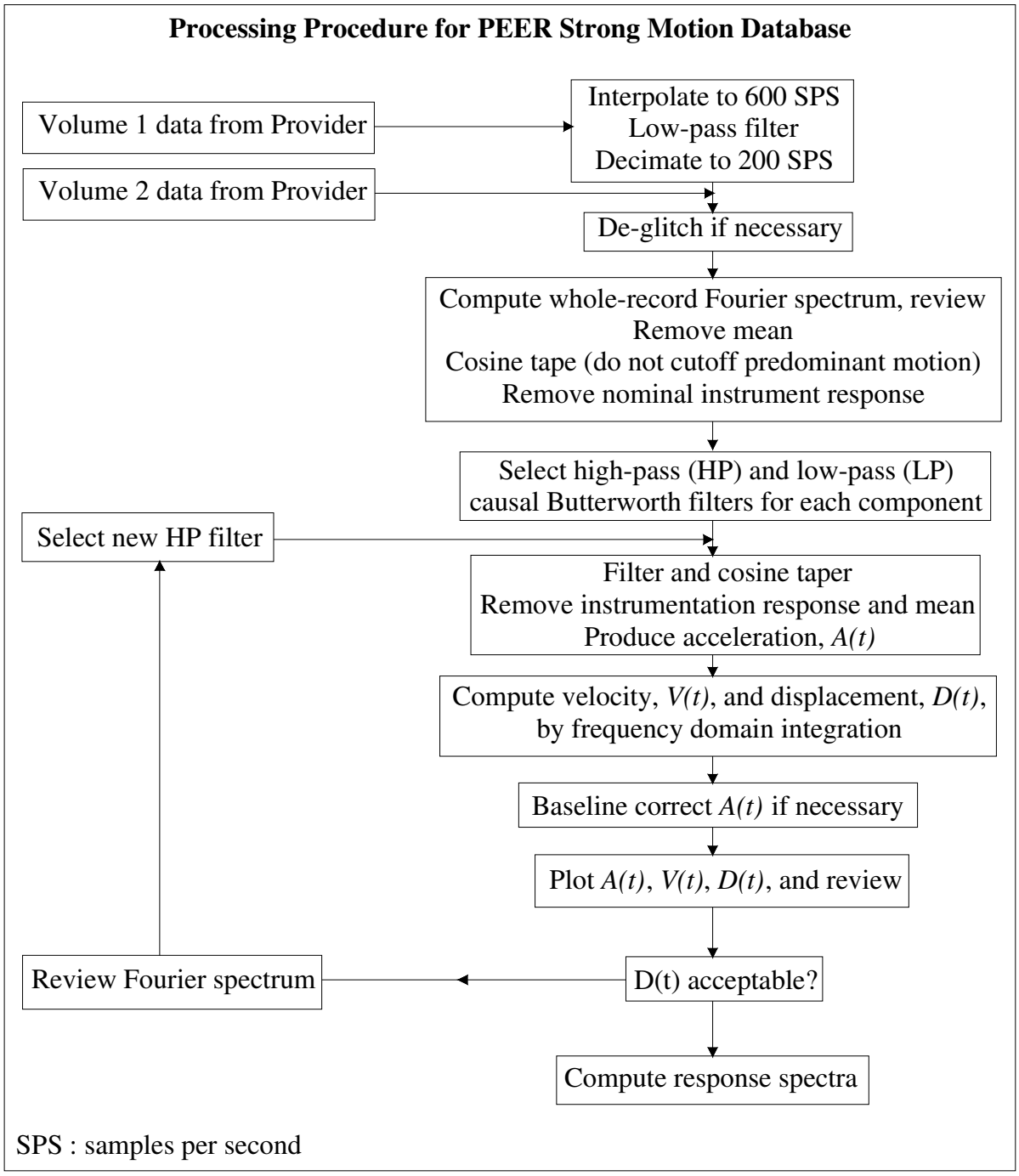


Figure 3.1. Processing procedure for PEER strong motion database

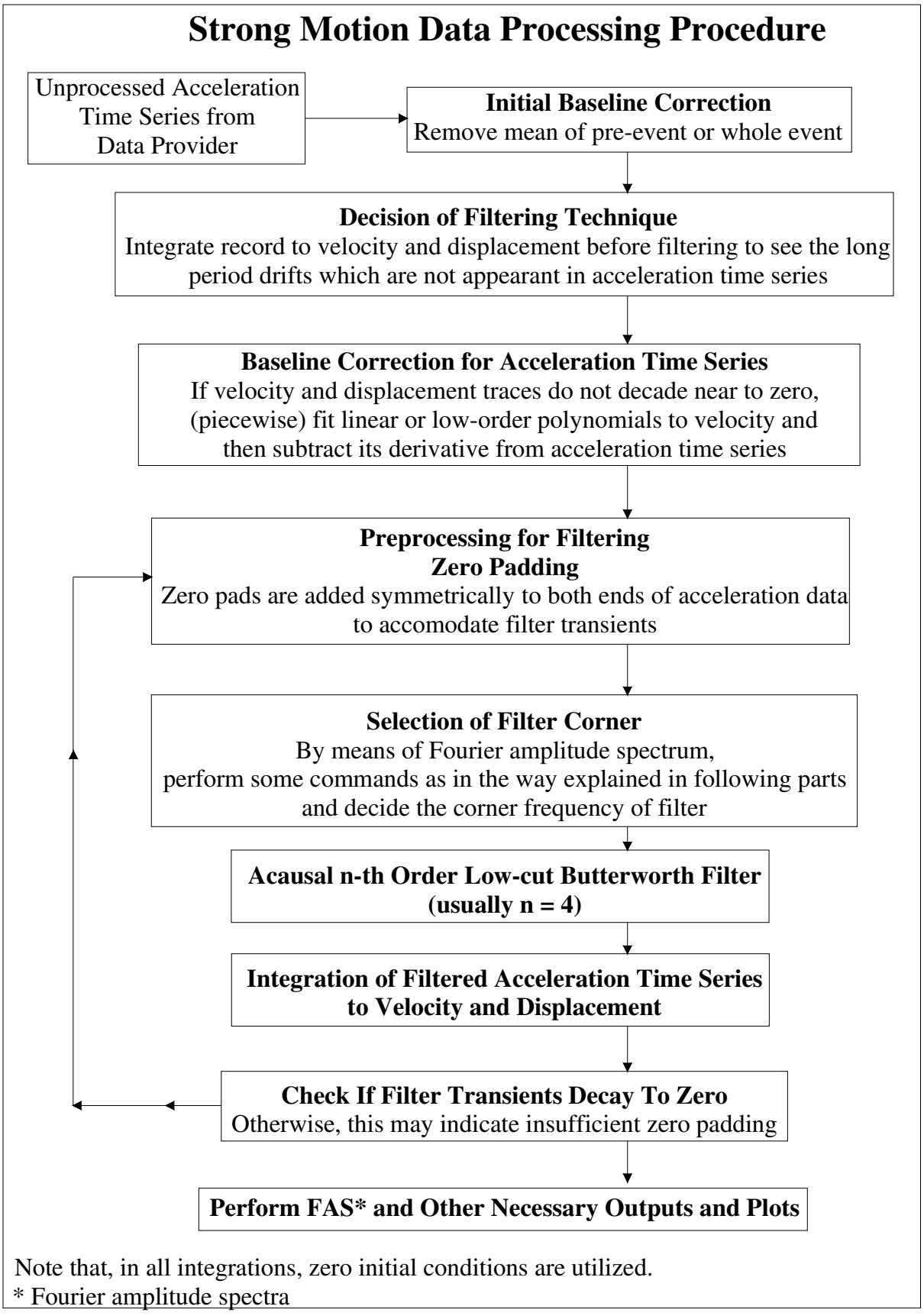


Figure 3.2. General form of processing procedures

3.7. Baseline Corrections

Baseline problems result in unphysical velocities and displacements. Before the baseline correction, the mean of a suitable portion of the acceleration record, either the interval prior to the P-wave (pre-event signal) or the mean of the entire record is subtracted from the whole record. After making the initial baseline correction by subtracting this mean, the record is integrated without filtering. The obtained velocity is checked for long-period drifts that may indicate the presence of offsets in the reference baseline.

The zero acceleration level is called baseline or centerline. Some difficulties in determining the baseline position are as follows:
initial part of shock and sometimes final part of the shock is not recorded,

1. final accelerations or velocities do not tend to zero, due to the presence of background noise,
2. the final displacement is not known.

In most cases, baseline offsets are small, and simply filtering the data can effectively remove this noise. For cases of which the baseline offsets are considered large enough and damage the signal at long periods, there exist certain number of solutions proposed as baseline correction in the literature. Some of them are :

1. corrections as piecewise fitting of linear or low order polynomials to the velocity parabolic acceleration baseline (cubic baseline on the velocity). Then subtracting the derivative of these fits from the acceleration time series are made.
2. three parts of the acceleration baseline which have different zero levels (that before the strong motion, that during the strong motion and that after the strong motion)

3.8. Filtering

A filter is a function that in the frequency domain has a value close to 1 in the range of wanted frequencies and close to zero in the range of frequencies that the analyst wishes to eliminate. A filter is defined by a filter frequency and an order: the higher the order of the filter, the more rapid the roll-off. (As order of a filter increases, the slope gets steeper).

Low-pass filters offer difficult passage to high frequency (short period) signals while allowing passage of low frequency signals. For low-pass filters, one can come across with the

term high-cut indicating the frequencies being removed. High-pass filters blocks signals with frequencies that are too low. Between these two, band-pass filters pass a limited range of frequencies. A first order filter has only a single frequency-dependent component.

A causal filter depends only on the current and previous inputs. The poles of the digital filter are within the unity circle which indicates stability of the system (with negative exponentials).

On the other hand, an anticausal (acausal) filter has positive exponentials and gets values from both the past and the future. Signals whose poles are mapped outside the unity circle are exponentially growing signals. This is a term often used in control theory and digital signal processing. An n-th order filter can be decomposed into the summation of subfilters as causal and acausal one.

3.8.1. Choice of Filtering Technique

The filter can be applied in the time domain, by convolution of its transform with the time history, or in the frequency domain by multiplying the filter function with the Fourier amplitude spectrum (FAS) of the time history, and then obtaining the filtered time history through the inverse Fourier transform. The choice between application in the time domain or the frequency domain is not important and exactly the same results should be obtained in both cases.

In order to remove the short and long period errors from accelerograms, there exist different types of filters. Some of them are Ormsby, Butterworth, Chebychev, Bessel and elliptical filters. However, type of the filter is not important. Correct application of the chosen filter is much more important than the application domain (time or frequency domain) and the choice of a particular filter.

The Butterworth filter is designed to have a frequency response which is as flat as mathematically possible in the passband. This is the reason of prefering Butterworth filter rather than the Bessel, elliptic or Chebyshev filter.

The key issue of choosing a filtering technique is whether the filter is causal or acausal. The distinguishing feature of acausal filters is that they do not produce any phase distortion in the signal, whereas causal filters result in phase shifts in the record. For routine processing, acausal filtering is preferred over causal filtering. One reason for this preference is that at periods much shorter than the corner frequency the waveforms, and hence the response

spectra, tend to be less sensitive to the low-cut-frequency corner. At periods much shorter than the filter corner the responses are significantly less sensitive to the corner period for acausal filtering, particularly for the inelastic response. (Boore and Akkar)

3.8.2. Zero Padding

Before applying the low-cut-frequency filter, zero-pads are added symmetrically to both ends of the records in order to accommodate the filter transients. These pads are needed regardless of whether the filtering is done in the time-domain or in the frequency-domain. The length of the pads depends on the filter frequency and the filter order. The required length of the filter pads will often exceed the usual lengths of pre-event and post-event memory on digital recordings, hence it is not sufficient to rely on the memory to act as the pads. The length of the pad at each end, t_{pad}, is determined by this empirical formula:

$$t_{pad} = 1.5 * \frac{nroll}{f_c}$$

$$nroll = \frac{n}{4}$$

where, *nroll* is the rolloff of the acausal Butterworth filter, n is the filter order, *fc* is the corner frequency of the low-cut-frequency filter in hertz.

A decay near to zero at the ends of the velocity and displacement traces which are obtained by multiple integrations of the filtered accelerations may show that the applied padding is adequate.

3.8.3. Low-pass Filters (High-cut Filters)

If it is decided to apply a filter due to the presence of significant high-frequency noise in the record, then filter can be applied in the frequency domain or the time domain. The upper frequency limit of the usable range of high frequencies in the motion is determined by the Nyquist frequency. It is the highest frequency at which characteristics of the motion can be correctly determined and is equal to ½ dt where dt is the sampling interval. A high-cut filter applied at frequencies greater than the Nyquist will have no effect on the record.

The transducer frequency in analog instruments is limited to about 25 Hz. This results in distortions of amplitudes and phases of the components of ground motion at frequencies close to or greater than that of the transducer. The digitization process itself can also introduce high-frequencies. Very high-frequency motions will also tend to attenuate rapidly with distance and therefore will not be observed at stations even a few tens of kilometers from the fault rupture.

3.8.4. Low-cut Filters

The low-cut filter is the most effective and most commonly used way of reducing the long-period noise in accelerograms. The improvement in the appearance of velocity and displacement time histories by the application of filter to the acceleration time history can be recognised easily, even if a very slight difference between the filtered and unfiltered accelerations exist.

The goal of a low-cut filter is removing that part of the signal that is judged to be heavily contaminated by long-period noise. The key issue is selection of the period beyond which the signal-to-noise ratio is unacceptably low.

The LP and HP filter frequencies are selected individually for each component of a record based on an assessment of the Fourier amplitude spectrum and the integrated displacement time history.

3.9. Fast Fourier Transform (FFT) and Fourier Amplitude Spectrum (FAS)

The Fourier transform is a reversible integral transform of one function into another. The Fourier transform gives the coefficients of sinusodial basis functions. The linear combination of these functions, which is performed by summation or integration produces the original function. The inverse Fourier transform recombines these sinusodial basis functions.

The fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT) and its inverse. The discrete Fourier transform (DFT) is defined by the following formula

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk} \qquad k = 0,\ldots, N-1$$

where x_0, …., x_{N-1} are complex numbers.

The fast Fourier transform (FFT) reduces the number of computations needed for *N* points from N^2 to $N \log N$.

In terms of signal processing, a time series representation of a signal function is mapped into frequency domain by the Fourier transform. It is a decomposition of a function into harmonics of different frequencies. In Appendix 11, the magnitude of the resulting complex-valued function fft_mag represents the amplitudes of the respective frequencies, while the phase shifts are given by arctan (imaginary parts/real parts), but phase shifts are out of the thesis's interest.

CHAPTER 4

PROCESSING OF EARTHQUAKES

4.1. Selection of Acceleration Data

There exists no certain way of record processing. Generally it is not possible to identify the 'best' processing for an individual record: assumptions always need to be made and the optimal procedure for a given record will depend on the application and the record itself. The limitations of the data and the processing routines need to be appreciated by the end users.

Earthquakes travel at a finite speed of 250-800 m/s and are modified by the ground through which they pass. The reliability of the record for exhibiting the real nature of strong ground shake depends primarily on the distance to the epicenter of earthquake.

The records utilized in this work are selected due to their maximum acceleration values and magnitudes given in Table 4.1. These records are marked by bold characters in the table. Maximum acceleration values of some other important earthquakes recorded by National Strong Ground Motion Record Network of Turkey (TKYHP) are also given in Table 4.1. The following abbreviations are used throughout the current chapter:

N-S North-South direction,
E-W East-West direction,
V vertical direction,
G ground acceleration unit, gal,
(1 gal = 1 cm/s^2)
mG miligal (10^{-3} gal),
S-P time difference of arrival to the station of P and S waves,
AVD acceleration, velocity, and displacement,
VD velocity and displacement,
ML local magnitude,
Mb body-wave magnitude,

Ms surface-wave magnitude,

Mw seismic moment magnitude,

HP high-pass (low-cut) filter,

LP low-pass filter.

Table 4.1. Maximum acceleration values of some important earthquakes recorded by National Strong Ground Motion Record Network of Turkey (TKYHP)

Date	Time (GMT)	Epicenter Coordinates	Magnitude	Distance (km)	Station	Max Acceleration (gal)
19.08.1976	01:12:40	37.71K - 29.00D	5.0 Md	15.1	DENİZLİ	348.5
18.09.1979	13:12:23	39.66K - 28.65D	5.2 Md	67.6	Dursunbey(BALIKESİR)	288.2
18.09.1979	13:12:23	39.66K - 28.65D	5.2 Md	67.6	Dursunbey(BALIKESİR)	288.2
30.06.1981	07:59:09	36.17K - 35.89D	4.7 Md	24.6	HATAY	154.0
30.10.1983	04:12:28	40.20K - 42.10D	6.8 Ms	18.5	Horasan (ERZURUM)	173.3
05.05.1986	03:35:38	38.02K - 37.79D	5.8 Ms	29.6	Gölbaşı (MALATYA)	114.7
13.03.1992	17:18:39	39.72K - 39.63D	6.8 Ms	10.4	ERZİNCAN	470.9
01.10.1995	15:57:13	38.11K - 30.05D	5.9 Ms	10.8	Dinar (AFYON)	329.7
27.06.1998	13:55:53	36.85K - 35.55D	5.9 Ms	30.0	Ceyhan (ADANA)	273.5
17.08.1999	00:01:51	40.70K - 29.91D	7.4 Mw	40.0	SAKARYA	407.0
12.11.1999	16:57:20	40.74K - 31.21D	7.2 Mw	12.6	DÜZCE	513.7
12.11.1999	**16:57:22**	**40.74K - 31.21D**	**7.2 Mw**	**33.4**	**BOLU**	**805.9**
06.06.2000	10:41:40	40.63k - 33.03D	5.9 Md	23.9	Çerkeş (ÇANKIRI)	63.2
02.03.2002	07:11:29	38.46K - 31.30D	6.1 Md	73.9	Afyon	113.4
01.05.2003	**00:27:08**	**39.01K – 40.47D**	**6.4 Mw**	**10.5**	**Bingöl**	**545.5**

4.2. 01 May 2003 Bingöl Earthquake (Mw = 6.4)

Data and station information are given in Appendix 4 and Appendix 5. The record of 01 May 2003 Bingöl earthquake from Bingöl station contains pre-event signal of 20 seconds.

The recorder type is GSR16. The number of data is 6474 and sample interval is 0.01 seconds. The site conditions of the station are given in Appendix 4.

The peak accelerations in north-south, east-west and vertical directions are, respectively, 545.5326, 276.8251 and 472.2599 gal (=cm/s^2). The units given in header part of unprocessed acceleration data file from ERD and pictures of records given in Appendix 3 are mG (miligal). This is confusing, but it is much more logical that the unit of accelarations are gal (=cm/s^2). The processed data obtained from PEER also support this fact. ERD is the data provider of PEER, but there should be something wrong with header part of ERD data. This error is only exists in header part of data file, the acceleration data are correct. The unit of unprocessed data obtained from ERD can be accepted as mg (thousandth of ground acceleration) instead of mG(miligal).

"The epicenter of the main shock was located to the north of Bingol, a city that is surrounded by a set of very complex and heterogeneous fault patterns. On the macroseismic scale, the earthquake occurred inside the Bingol-Karliova-Erzincan triangle that is confined by the Karliova triple junction from the east, the right lateral strike-slip North Anatolian fault (NAF) from the north and left lateral strike-slip East Anatolian fault (EAF) from the south. The Bingol-Karliova-Erzincan triangle is confined and traversed by conjugate faults of the NAF and EAF that run in the NE-SW and NW-SE directions. The station is located in the north of the city, on an estimated 50 m high alluvial terrace between two streams. The terrace material is dense formations comprising predominantly uniform granular alluvial deposits. Further north from the building, there are slopes toward the bottom of the valley formed by the second stream. The authors do not have an upper 30 m shear wave velocity variation of the area of interest but the geological formation described above suggests that the soil profile in this area would be classified as USGS site class C or D (i.e., 360 £ vs < 750m/s and 180 £ vs < 360m/s, respectively). " (Özcebe at al.)

All components of maximum acceleration records from ERD are given in Appendix 3. P and S waves are marked on the accelerograms. The magnitude and depth of the earthquake are 6.4Mw and 6.0 km. The epicenter distance is 10.5 km. The station record utilized among the other main shock records is indicated by bold characters in Table 4.2.

Table 4.2. 01 May 2003 Bingöl earthquake 00:27 (Mw = 6.4) main shock records

File number	Date	Time (GMT)	N-S (mG)	E-W (mG)	V (mG)	S-P (sec)	Instrument	Epicenter	Station
200305010027-TAT	01.05.2003	00:27:04	5.98	4.24	2.90	22.66	GSR-16	Bingöl	TAT
200305010027-ERC	01.05.2003	00:27:04	8.34	7.50	4.11	16.36	SSA-2	Bingöl	ERC
200305010027-TER	01.05.2003	00:27:04	5.10	10.30	4.30	12.28	GSR-16	Bingöl	TER
200305010027-KMR	01.05.2003	00:27:04	1.50	1.50	1.00	N/A	SM-2	Bingöl	KMR
200305010027-MLZ	01.05.2003	00:27:04	5.00	5.50	3.00	N/A	SM-2	Bingöl	MLZ
200305010027-ELZ	01.05.2003	00:27:04	8.00	7.00	5.00	16.24	SM-2	Bingöl	ELZ
200305010027-BNG	**01.05.2003**	**00:27:04**	**545.53**	**276.82**	**472.26**	**1.55**	**GSR-16**	**Bingöl**	**BNG**

Fourier amplitude spectrum of north-south and east-west components of raw acceleration records are given in Fig.4.4. The integrated velocity and displacement traces of north-south and east-west components of raw acceleration data are shown in Fig.4.1. The filter corner frequencies can be determined by means of Fourier amplitude spectrum of raw accelerations and VD traces of raw accelerations.

The unprocessed velocities show a linearly increasing trend which results in a higher order polynomial in displacements. The end values of velocity and displacement traces do not tend to zero. This fact is much more visible in displacements and indicates that raw acceleration data needs to be processed.

As the starting point of the correction procedures, mean correction (zeroth order correction) is applied to the raw acceleration data. This is performed by removing the mean of the pre-event signal from the raw record. The mentioned data which belong to Bingöl station has a pre-event signal of 20 seconds.

Acausal filtering is prefered to causal one due to the fact that response spectra can be sensitive to the corner frequencies in causal filtering. As preprocessing for acausal filtering, zero pads are added to beginning and end of the record. Pre- and post-zero pads are required for acausal filters. The initial length of the record was 64.74 s. After the application of zero padding it artificially elongates as 120.74 s, where the additional 20 s are leading and 40 s are trailing zeros.

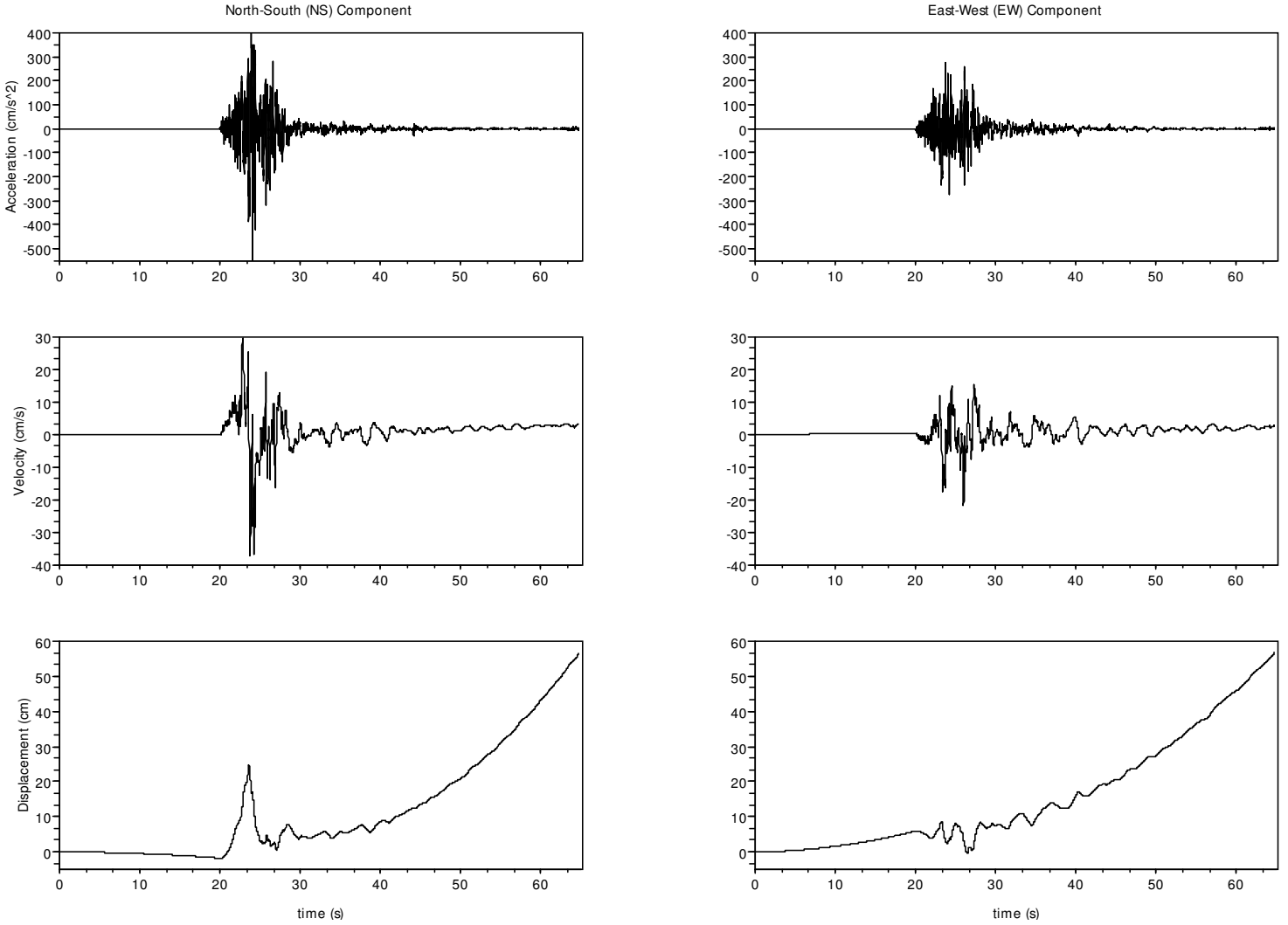


Figure 4.1. VD traces obtained by integration of raw acceleration data of NS and EW components (01 May 2003 Bingöl earthquake 00:27 Mw=6.4 Bingöl station)

A first order low-cut acausal Butterworth filter with a corner frequency (fc) of 0.04 Hz was used. The results are given in Fig.4.2 as accelerations, velocities and displacements of north-south (NS) and east-west (EW) components of Bingöl station. This processing procedure is the one applied in (Özcebe at al.). The resultant displacements in (Özcebe at al.) are given in Fig.4.3. These graphs are copied from the pdf file of the relevant reference. In the context of the thesis, alternative 2 with gray line is applied. The results obtained from comparison of displacements in Fig.4.2 and in Fig.4.3 are satisfactory. Some slight differences exist due to the different integration schemes, the application of different filter orders, and zero-padding. In Fig.4.3., the precursory motion of the displacements that are derived from filtering is due to the leading zeros required in acausal filtering.

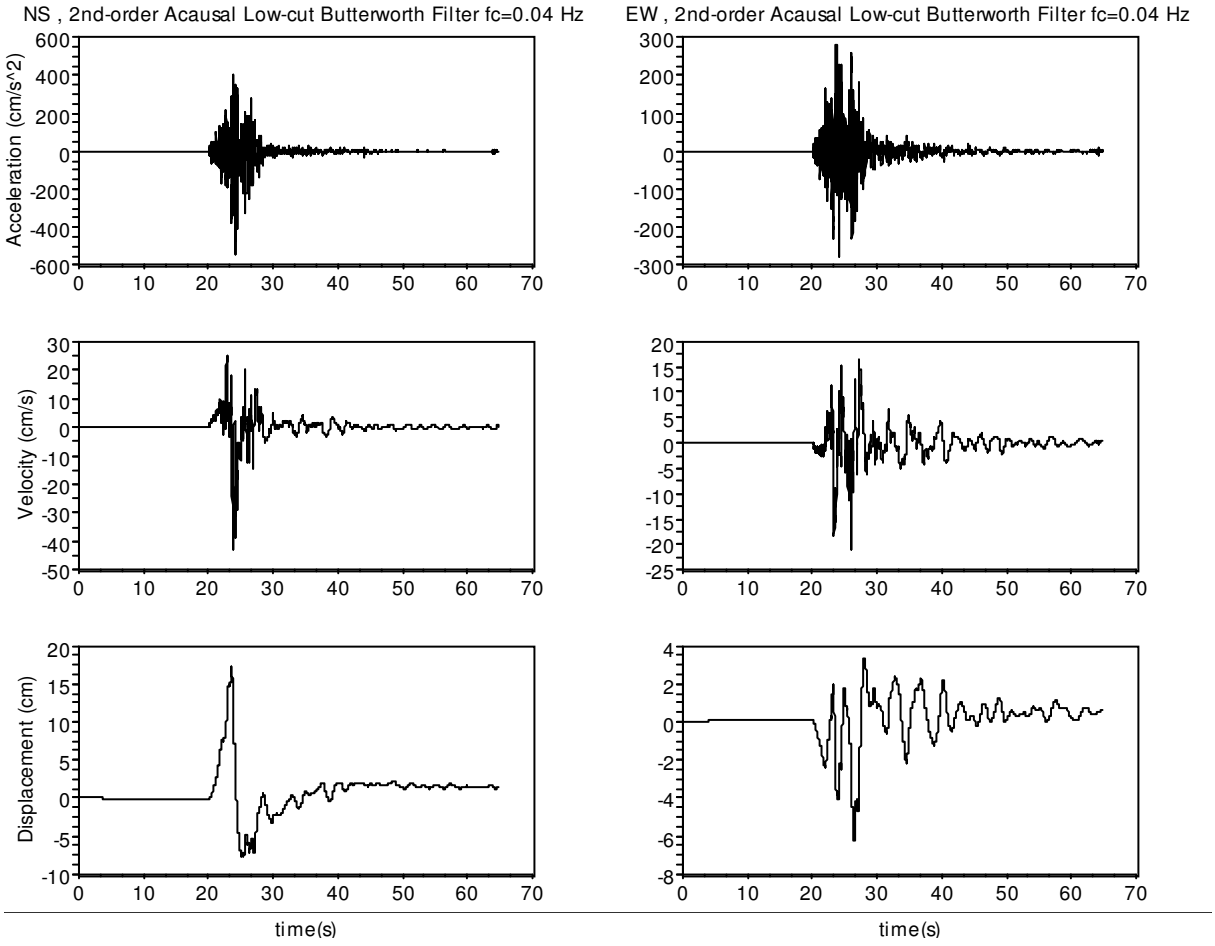


Figure 4.2. Processed AVD traces of NS and EW components (01 May 2003 Bingöl earthquake 00:27 Mw = 6.4 Bingöl station)

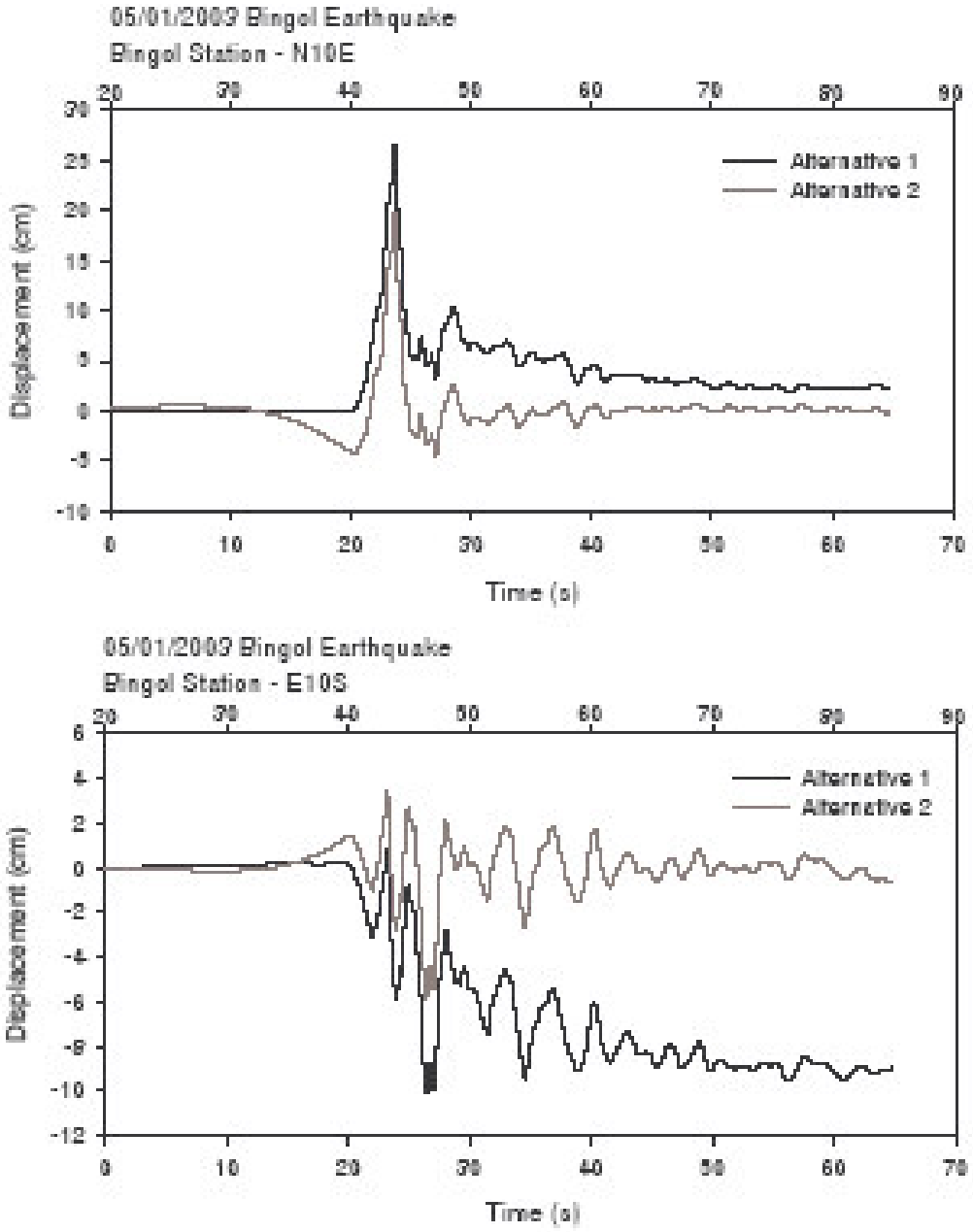


Figure 4.3. AVD traces of NS and EW components processed by (Source : Özcebe at al.)
(01 May 2003 Bingöl earthquake 00:27 Mw = 6.4 Bingöl station)

Fourier amplitude spectrum of processed acceleration data of north-south components and easth-west components are given in Fig.4.4. The parts of records that are filtered can be easily seen at low frequency region.

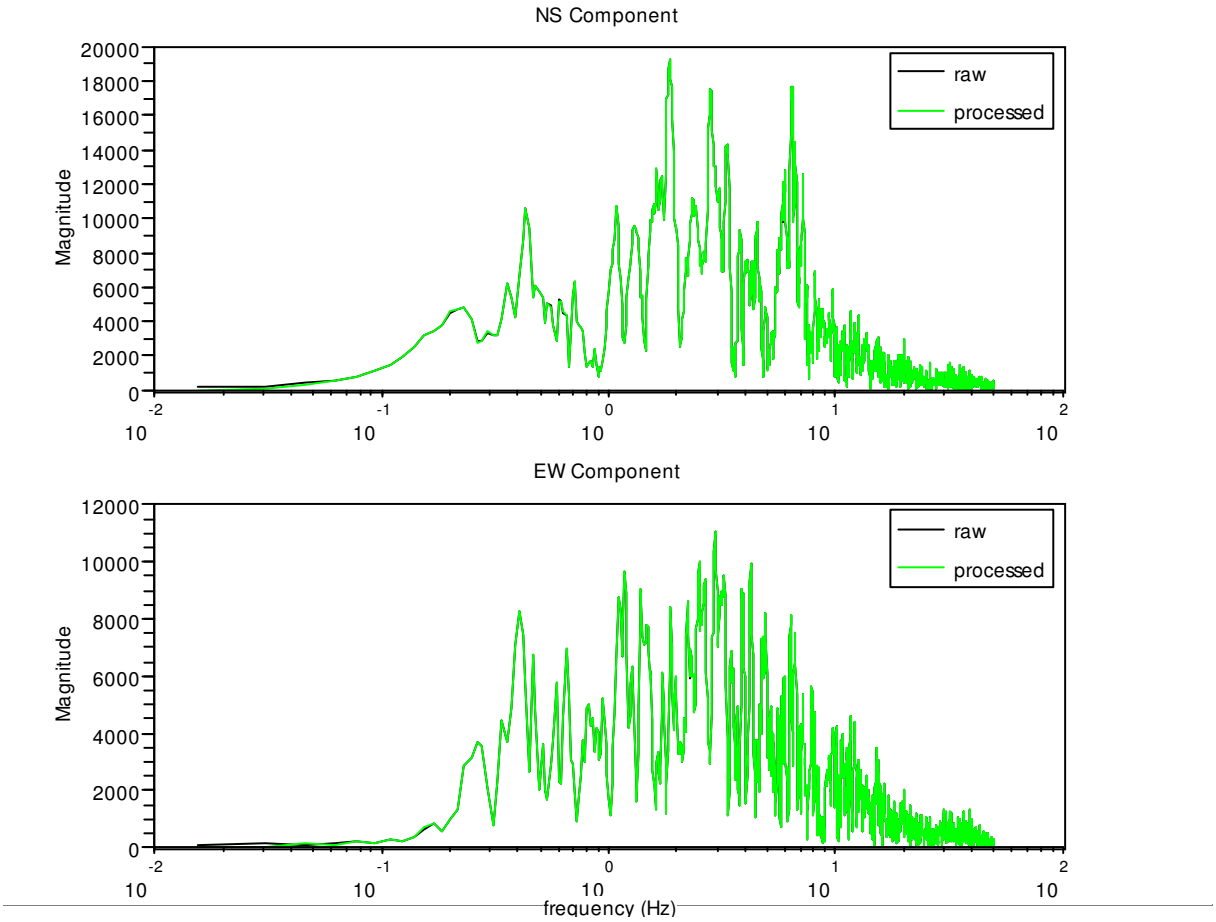


Figure 4.4. FAS of acceleration time series (1st order acausal low-cut Butterworth filter with fc = 0.04 Hz) (01 May 2003 Bingöl earthquake 00:27 Mw=6.4 Bingöl station)

4.2.1. Effect of Zero Padding

In lack of zero-pads or in case of inefficient length of zero-pads, the end values of velocity and displacement traces do not tend to zero. This is generally more obvious in displacement traces. The effect of inefficient length of zero-pads can be easily seen in Fig.4.5. The displacement data are obtained by twice integration of acceleration data which have been filtered by a second order acausal low-cut Butterworth filter.

At the end of the ground shaking caused by an earthquake, the ground velocity must return to zero and this is indeed a criterion to judge the efficiency of the record processing. The final displacement, however, need not be zero since the ground can undergo permanent deformation. But in this case it is not a permanent deformation, but a deflection coming from the procedure that is being used. This can be understood easily from the shape of displacement trace.

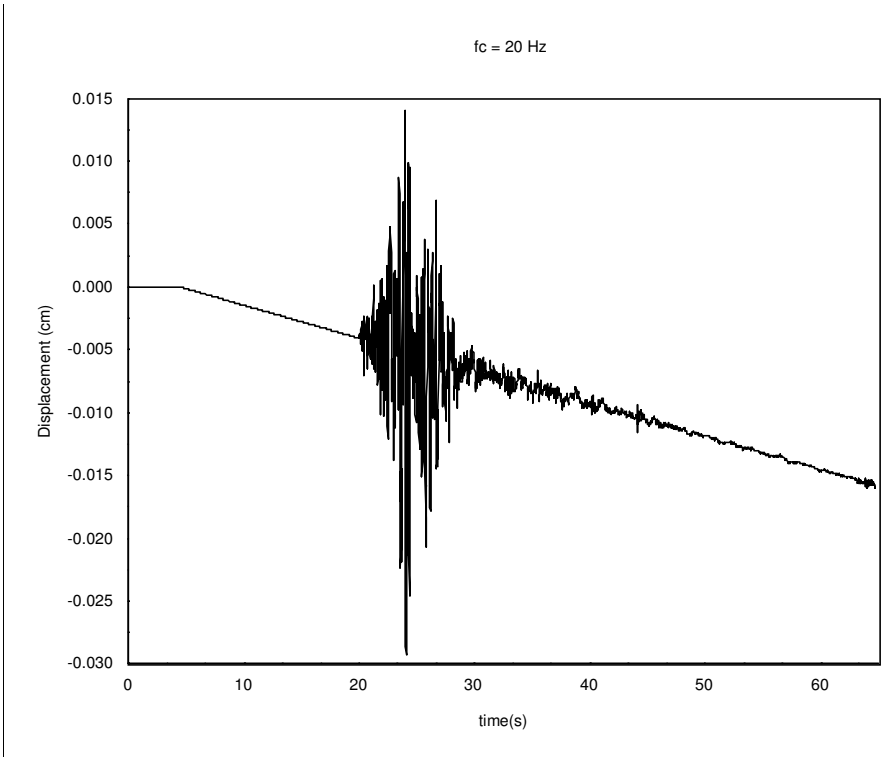


Figure 4.5. Effect of inefficient length of zero-padding (01 May 2003 Bingöl earthquake 00:27 Mw=6.4 NS component, 2nd order acausal low-cut Butterworth filter)

4.2.2. Effect of Corner Frequency

Some calculations are performed to visualize the effect of applying filters with different corner frequencies. The magnitudes and shapes of the velocity and displacement traces are completely different as they can be seen in Fig.4.6. Three different corner frequencies are utilized which are 1 Hz, 5 Hz, and 10 Hz, respectively.

The FFT magnitudes of acceleration data are given in Fig.4.7. The parts of records that are filtered and the parts that are taken into account by filtering can be easily seen.

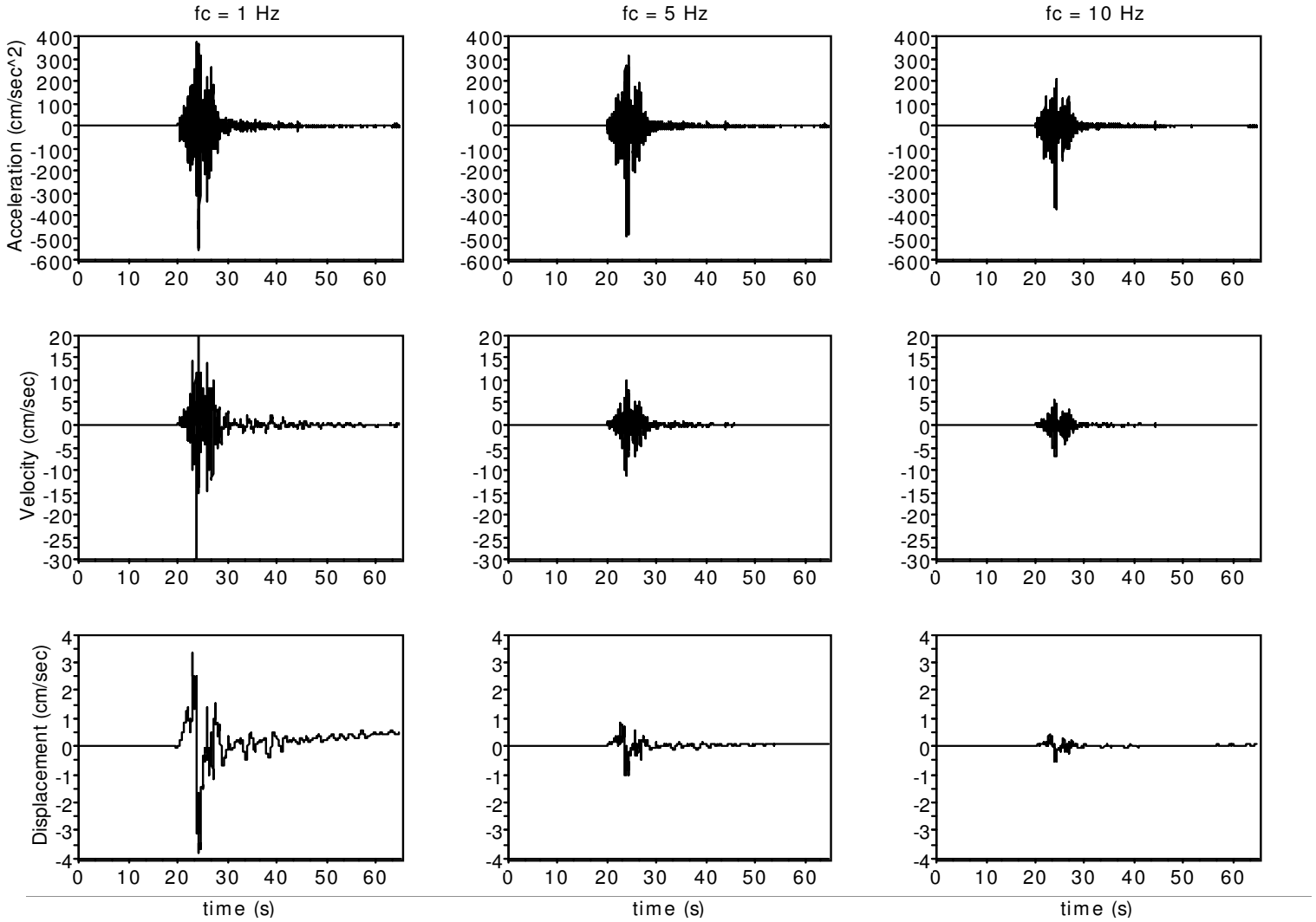


Figure 4.6. Effect of different filter corner frequencies (1st order acausal low-cut Butterworth filter) (01 May 2003 Bingöl earthquake 00:27 Mw=6.4 Bingöl station NS Component)

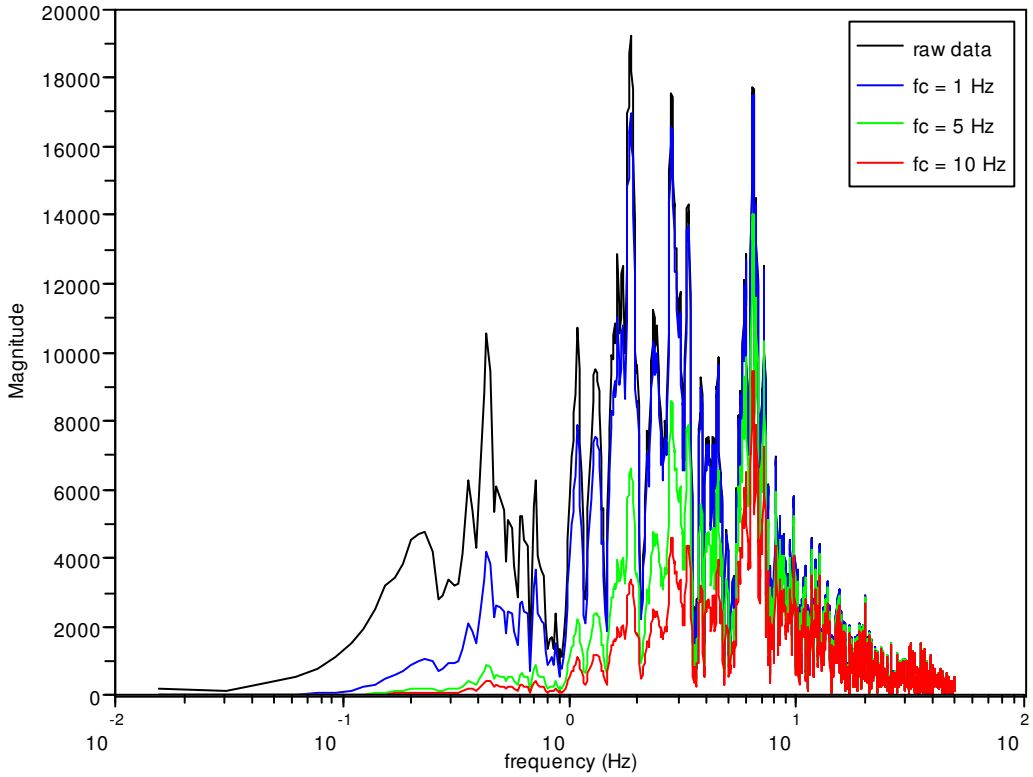


Figure 4.7. FAS of acceleration time series filtered with different corner frequencies (1st order acausal low-cut Butterworth filter) (01 May 2003 Bingöl earthquake 00:27 Mw=6.4 Bingöl station NS Component)

4.3. 12 November 1999 Düzce Earthquake (Mw = 7.2)

Data and station information are given in Appendix 7 and Appendix 8. The record of 12 November 1999 Düzce earthquake from Bolu station contains its pre-event signal of 5 seconds. The recorder type is GSR18. The number of data is 5590 and sample interval is 0.01 seconds. The site conditions of the station are given in Fig. A.7.

The peak accelerations in north-south, east-west and vertical directions are 739.5120, 805.8780 and 200.1300 gal (=cm/s^2), respectively. All components of maximum acceleration records from ERD are given in Appendix 6. The magnitude and depth of the earthquake are 7.2 Mw and 25.0 km. The epicenter distance is 33.4 km. The main shock records are given in Appendix 9. The record of Bolu station is utilized among the other main shock records.

PEER is the processed data provider and ERD is the raw data provider of the thesis. The origin of raw data of PEER and of the thesis are same. This fact is explained to show the compatibility of raw and processed data.

The unprocessed displacements obtained by integration of raw accelerations are given in Fig.4.8.

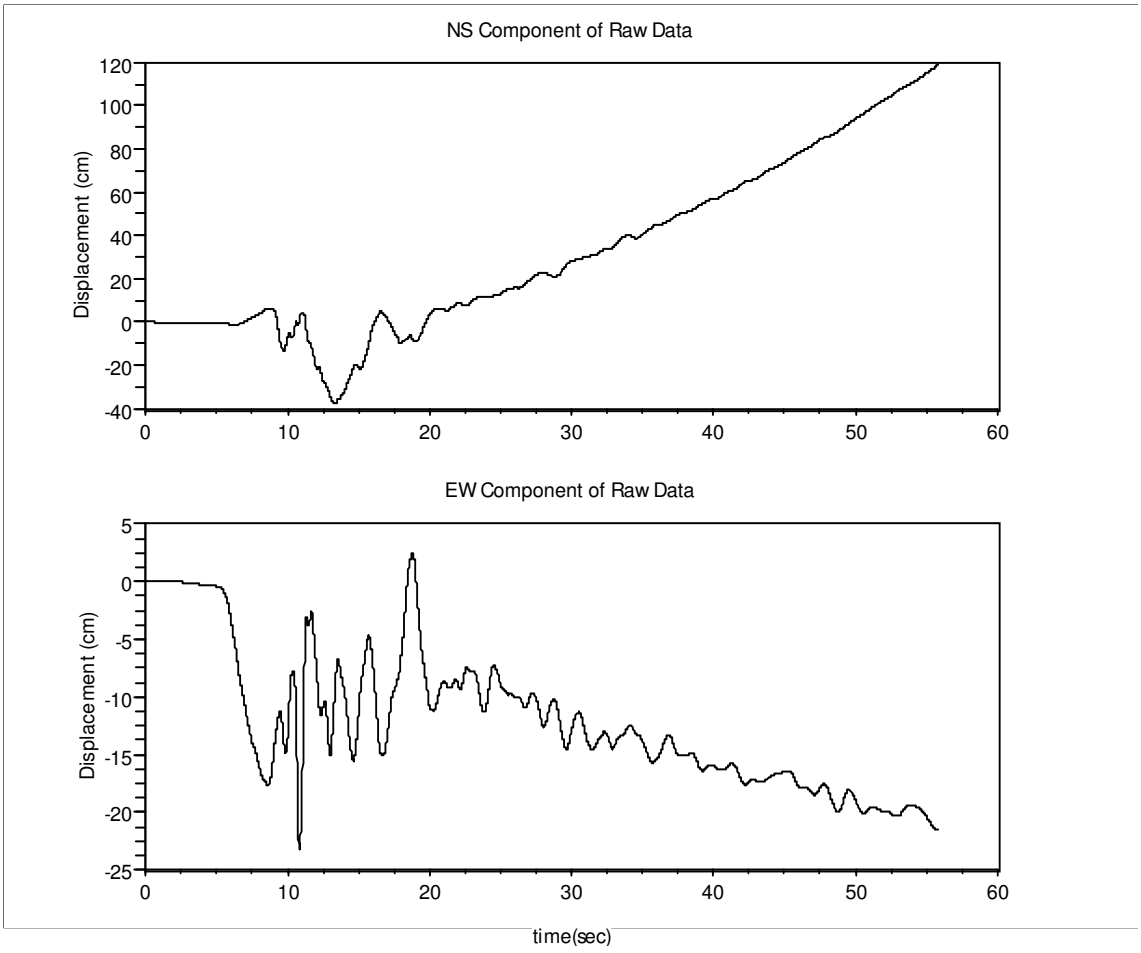


Figure 4.8. Displacement traces obtained from raw accelerations (12 November 1999 Düzce earthquake 16:57 Mw=7.2)

The green values in Fig.4.9. and Fig.4.10 are the ones processed by the author with parameters given by PEER. The information about PEER's data is given in Table 4.3. A 1st order low-cut acausal Butterworth filter with a corner frequency of 0.05 Hz is applied. Zero-pads are added as 7.5 s of leading and 7.5 s of trailing zeros.

The results are satisfactory. Slight differences may come from different numerical integration schemes, different filter orders, length of zero-pads, and from the fact that PEER

utilizes causal Butterworth filter. On the other hand, in this study, acausal Butterworth filter is applied.

Table 4.3. Data Information from PEER
(12 November 1999 Düzce earthquake 16:57 Mw = 7.2)

Record / Component	**HP (Hz)**	**LP (Hz)**	**Peak Ground Acceleration PGA (g)**	**Peak Ground Velocity PGV (cm/s)**	**Peak Ground Displacement PGD (cm)**
DUZCE/BOL-UP	0.05	null	0.203	17.3	14.29
DUZCE/BOL000	0.05	null	0.728	56.4	23.07
DUZCE/BOL090	0.05	null	0.822	62.1	13.55

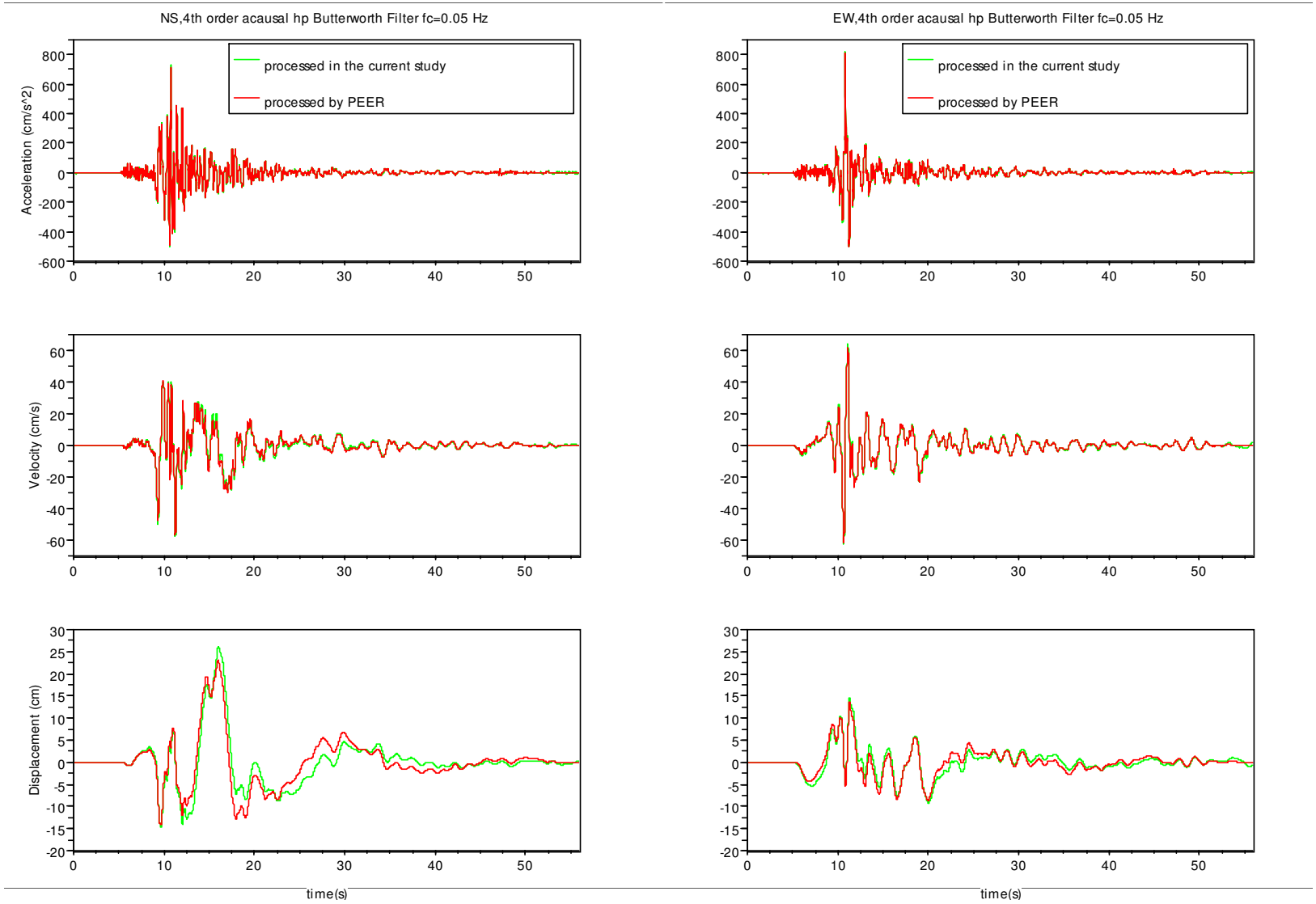


Figure 4.9. Processed AVD time series (12 November 1999 Düzce earthquake 16:57 Mw=7.2)

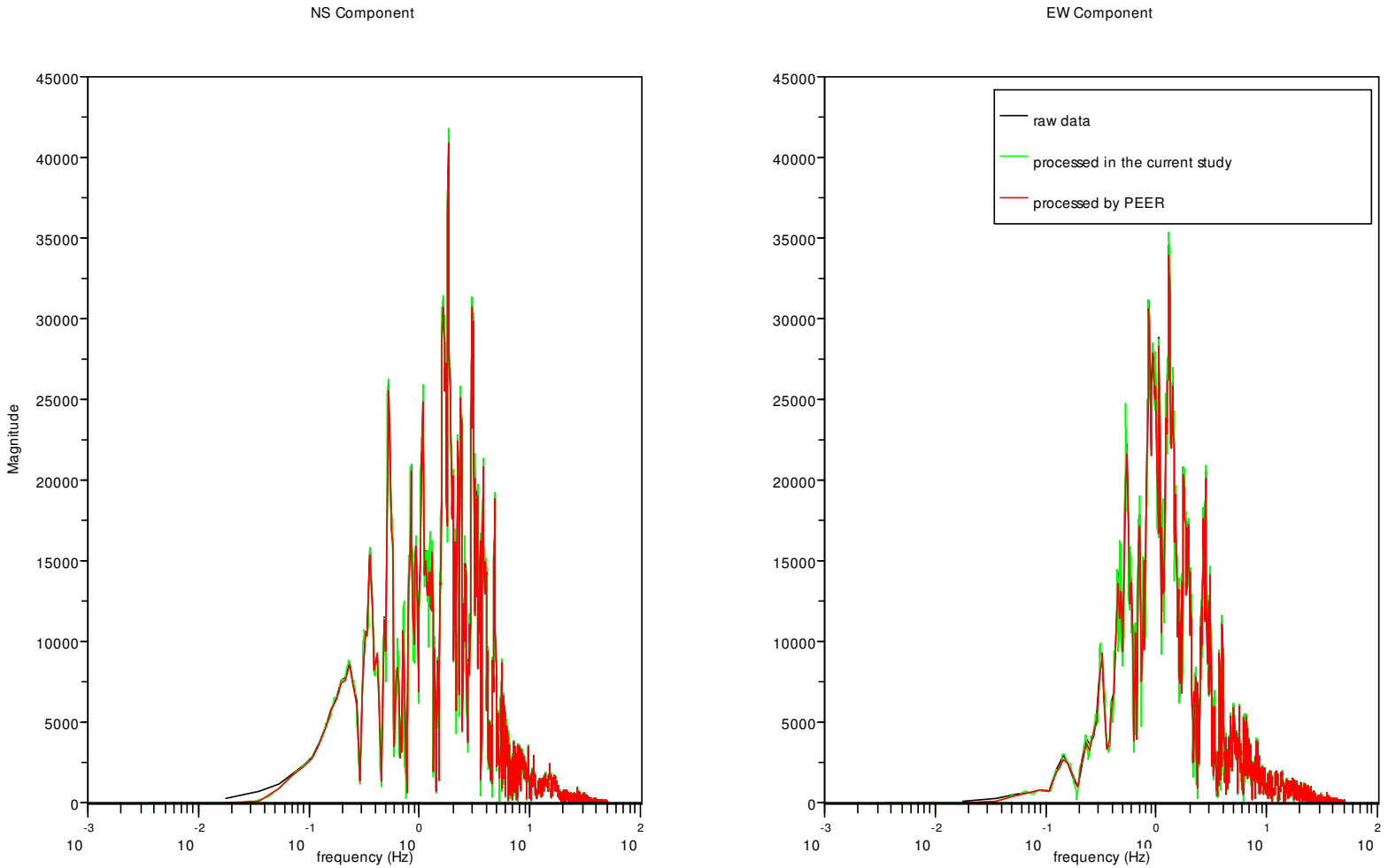


Figure 4.10. Fourier amplitude spectrum of different acceleration data (12 November 1999 Düzce earthquake 16:57 Mw=7.2)

4.4. Conclusions

In the context of the present study, the accelerograms of two major earthquakes in Turkey are processed. Comparison of results obtained in the present work and from two references are satisfactory. Some slight differences exist due to the different numerical integration schemes, the application of different filter orders, length of zero-padding, and from the fact that PEER utilizes causal Butterworth filter while acausal Butterworth filter is applied in this study.

Effects of different factors like corner frequencies and zero pads are studied and are graphically represented.

CHAPTER 5

CONCLUSION

The goal of this work is simulation of recorded earthquakes in a laboratory environment. For this reason, the simulations are performed in a regular Linux environment and also in a realtime Linux environment. The advantage of the realtime environment ensures the signals send to the servo driver to be on-time. Any time delay due to operating system tasks can not occur in a realtime Linux simulation and the frequency content of earthquake shaking can be reflected correctly during the simulation.

SHAKE1 will be utilized in the civil engineering department of IYTE, in earthquake and structural dynamics lectures to visualize the dynamic response of any model structure and in presentations arranged with civilian associations to demonstrate earthquake effect.

Difficulties occurred during assembly of hardware and while working on non-realtime and realtime software. Problems associated to the usage of the DAQ card were left unanswered by the manufacturer. Thus, causing times of frustration during the coding process.

The second part of the present study focused on the strong motion data processing. Two earthquakes are processed. These are 2003 Bingöl and 1999 Düzce earthquakes. The former one is processed as in the way explained in (Özcebe at al.) and the other one is processed as applied by PEER. The comparison of results are satisfactory. Some slight differences exist due to the different integration schemes and due to the application of different filters, zero-padding, and different filter orders.

5.1. Proposed Future Work

This is a starting step for larger shake tables. Future works in this area may be manufacturing of shake tables larger in dimensions with one-degree-of-freedom or may be superior in degrees-of-freedom.

The noise level in signals read from the acceleration sensor is moderately high. Filter application will be tried to reduce the noise.

REFERENCES

Ammanagi, S., Poornima, V. and Sera, A. “Development of a digitally-controlled 3-axis earthquake shake table”, TestResources Asia Pacific. India.

Benedetti, D., Carydis, P. and Pezzoli, P. 1998. “Shaking Table Tests on 24 Simple Masonary Buildings”, *Soil Dynamics and Earthquake Engineering*. Vol. 27, p. 67-90.

Boore, D.M. 2005. “On pads and filters: processing strong-motion data”, *Bulletin of the Seismological Society of America.* Vol. 95, No. 2, p.745-750.

Boore, D.M. and Bommer, J.J. 2005. “Processing of strong-motion accelerograms: needs,options and consequences”, *Soil Dynamics and Earthquake Engineering.* Vol. 25, p. 93.

Boore, D.M. and Akkar, S. 2003. “Effect of casual and acasual filters on elastic and inelastic response spectra” , *Earthquake Engineering and Structural Dynamics.* Vol. 32 , p.1729.

Boore, D.M., Stephens, C.D. and Joyner, W.B. 2002. “Comments on baseline correction of digital strong-motion data: Examples from the 1999 Hector Mine, California, Earthquake”, *Bulletin of the Seismological Society of America.* Vol. 92, No. 4, p.1543-1560.

Bucher, R. , Dozio, L. and Mantegazza, P. “Rapid control prototype with Scilab/Scicos and Linux RTAI” , Italy.

Bull, K.E., 2000. “Development of a shockwave tool for instruction of shake table operation”, Structural Engineering Research Experiences for Future Structural Engineers.

Converse, A.M. and Brady, A.G. “BAP : Basic Strong-motion Accelerogram Processing Software Version 1.0” , USGS , Open-file report 92-296A.

Çelebi, M., Akkar, S., Gulerce, U., Sanli, A., Bundock, H. and Salkin, A., “Main shock and aftershock records of the 1999 Izmit & Düzce, Turkey Earthquakes”, USGS/OFDA Project (No:1-7460-63170), Open-file report 01-163.

Darragh, B. , Silva, W. and Gregor, N. , “Strong –motion record processing for the PEER center” , PEER.

Douglas, J. , 2003. “What is a poor quality strong-motion record?” , *Bulletin of Earthquake Engineering.* Vol. 1, p.141-156.

Dyke, S. and Caicedo, J.M. “The University Consortium on Instructional Shake Tables” , Washington University in St.Louis.

Erdik,M., “Strong-motion data acquisition, processing and utilization in Turkey”, Bogazici University, Istanbul.

European Commission Earthquake Engineering Project report – Research Infrastructures

Gülkan, P., Akkar, S. and Yazgan, U.,2003. "1 Mayıs 2003 Sudüğünü (Sancak-Bingöl) Depremi Raporu- Bölüm II" , Ortadoğu Teknik Üniversitesi Mühendislik Fakültesi, Ankara.

Jugo, J. "Real-time controlof a DC motor using Scilab and RTAI", Bilboa, Spain.

Kuehn, J. , Epp, D. and Patent, W.N. 1999. "High-fidelity control of a seismic shake table" , Earthquake Engineering and Structural Dynamics,Vol. 28, p.1235.

Metronix User Manual (APD-VS Standard Series), Metronix.

Özcebe, G., Ramirez, J., Wasti, S.T. and Yakut, A. 2003. "1 Mayıs 2003 Bingöl Earthquake Engineering Report", TUBITAK.

PowerDAQ Programmer Manual, United Electronic Industries, Inc.

PowerDAQ User Manual, United Electronic Industries, Inc.

Real-Time Software Programmer Manual, SOAP Adaptive Module(SAM), Revision 2.0, May 2006

Reinhorn, A.M., Sivaselsan, M., Shao, Z.L.X., Pitman, M. and Weinreber, S. "Large scale real time dynamic hybrid testing technique – Shake tables substructure testing" , *Advances in Experimental Structural Engineering.*

Shakal, A.F. , Huang, M.J. , Graizer, V.M. , "CSMIP Strong-motion data processing" , CSMIP.

Skarlatoudis, A.A., Papazachos, C.B. and Margaris, B.N. 2003. "Determination of noise spectra from strong motion data recorded in Greece" , *Journal of Seismology.* Vol. 7 , p.533-540.

Skarlatoudis, A., Papazachos, B. and Margaris, B. "Recent advances in Greece on strong-motion networking and data processing" , Aristotle University of Thessaloniki.

Standard for the Exchange of Earthquake Data, Reference Manual, SEED Format Version 2.3., 1993.

Stephens, C.D. and Boore,D.M. "ANSS/NSMP strong –motion record processing and procedures" , USGS.

Trombetti, T.L. and Conte, J.P. 2002 . "Shaking table dynamics: results from a test-analysis comparison study" , *Journal of Earthquake Engineering.* Vol. 6, No.4 , p.513.

UCIST Shake Table Manual, Quanser Consulting Inc.

WEB_1, 2005. Scilab : A free scientific software package, 02/08/2005. http://www.scilab.org/

WEB_2, 2005. Harvard Seismology, 25/08/2005. http://www.seismology.harvard.edu/

WEB_3, 2005. EMSC European Mediterranean Seismological Centre, 25/08/2005. http://www.emsc-csem.org/

WEB_4, 2005. ESD European Strong Motion database, 27/08/2005. http://www.isesd.cv.ic.ac.uk/ESD/

WEB_5, 2005. International Seismological Centre, 27/08/2005. http://www.isc.ac.uk/

WEB_6, 2005. Scilab Notes and Functions, G.Urroz's web page, 30/08/2005. http://www.engineering.usu.edu/cee/faculty/gurro/Software_Calculators/Scilab_Docs/SCILAB_Notes&Functions.htm

WEB_7, 2005. Swarthmore E5 Engineering Methodology Lecture's web page, 01/09/2005. http://www.swarthmore.edu/NatSci/ceverba1/Class/e5/E5Matlab3/E5Matlab3.html

WEB_8, 2005. Seismological Station, Aristotle University of Thessaloniki, 02/09/2005. http://lemnos.geo.auth.gr/the_seisnet/en/index.htm

WEB_9, 2005. Sismoloji Şube Müdürlüğü, 02/09/2005. http://sismo.deprem.gov.tr/
Some Useful Linux Commands , http://www.er.uqam.ca/nobel/r10735/unixcomm.html

WEB_10, 2005. National Center for Research on Earthquake Engineering, 02/09/2005. http://www.ncree.gov.tw/eng/index.htm

WEB_11, 2005. National Observatory of Athens, 02/09/2005. http://www.noa.gr/indexen.html

WEB_12, 2005. The Cosmos VDC User Manual, COSMOS Strong-Motion Virtual Data Center, 05/09/2005. http://db.cosmos-eq.org

WEB_13, 2005. The George E. Brown Jr. Network for Earthquake Engineering Simulation, NEES at Berkeley, 05/09/2005. http://nees.berkeley.edu/Facilities/2.2.C.3.htm

WEB_14, 2005. Ergin Necati, C ve Sistem Programcılığı Derneği – C Ders Notları, 08/09/2005. www.nergin.com , http://www.csystem.org/downloads.php

WEB_15, 2005. Prof. Mete SÖZEN's web page, 20/10/2005. http://bridge.ecn.purdue.edu/~ce571/

WEB_16, 2005. METU EERC Earthquake Engineering Research Center, 03/12/2005. http://eerc.ce.metu.edu.tr/eng/index.php?id=6

WEB_17, 2005. METU SERU Civil Engineering Department Structural Engineering Research Unit, 03/12/2005. http://www.seru.metu.edu.tr

WEB_18, 2006. Anco Engineers, Inc., 10/02/2006. http://www.ancoengineers.com/shaketable.html, http://www.ancoengineers.com/servo.htm

WEB_19, 2006. Kandilli Rasathanesi ve Deprem Araştırma Enstitüsü, 19/02/2006. http://www.koeri.boun.edu.tr/sismo/

WEB_20, 2006. Bayindirlik ve İskan Bakanliği Afet İsleri Genel Müdürlüğü - Deprem Arastirma Dairesi, 08/03/2006. http://www.deprem.gov.tr/

WEB_21, 2006. Introduction to Digital Filters, Julius O. Smith, 10/04/2006. http://www-ccrma.stanford.edu/~jos/filters/

WEB_22, 2006. Introduction to DSP, BORES Signal Processing, 10/04/2006. http://www.bores.com/courses/intro/iir/

WEB_23, 2006. Prof. David BOORE' s web page, 20/04/2006. http://quake.wr.usgs.gov/~boore/

WEB_24, 2006. USGS-NSMP, 22/04/2006. http://nsmp.wr.usgs.gov/processing.html#notes

WEB_25, 2006. C Language Tutorial, 15/05/2006. http://www.physics.drexel.edu/courses/Comp_Phys/General/C_basics/c_tutorial.html#first

WEB_26, 2006. C#nedir?com, 23/05/2006. http://www.csharpnedir.com/

WEB_27, 2006. About, 17/06/2006. http://linux.about.com/od/commands/l/blcmdl.htm

WEB_28, 2006. Getting Started with RTLinux, 28/06/2006. http://piglet.uccs.edu/~chow/pub/rtl/doc/html/GettingStarted/

WEB_29, 2006. Ivchenko Alex, Application code and RTLinux, 28/06/2006. http://www.ueidaq.com/press/publications/appcodertlinux/

WEB_30, 2006. Ivchenko Alex, Get Those Boards Talking Under Linux (Part 1), 28/06/2006. http://www.ueidaq.com/press/publications/daqunderlinux1/

WEB_31, 2006. Ivchenko Alex, Get Those Boards Talking Under Linux (Part 2), 28/06/2006. http://www.ueidaq.com/press/publications/daqunderlinux2/

WEB_32, 2006. Ivchenko Alex, On The Move With Real-Time Linux, 28/06/2006. http://www.eetasia.com/ARTICLES/2002JUN/2002JUN01_ICD_CT_EDA_TA.PDF

WEB_33, 2006. Ivchenko Alex, Real-time Linux, 28/06/2006. http://www.embedded.com/story/OEG20010418S0044 http://www.commsdesign.com/design_corner/showArticle.jhtml?articleID=9900126

WEB_34, 2006. Ripoll I. "Real-time Linux (RT-Linux)", 30/06/2006. http://www.tldp.org/linuxfocus/English/May1998/article44.html

WEB_35, 2006. Linux Kitaplığı, 30/06/2006. http://www.belgeler.org/howto/

WEB_36, 2006. Cplusplus Resources, 30/06/2006. http://www.cplusplus.com/ref/

WEB_37, 2006. O'Reilly Linux Devcenter.com, 30/06/2006. http://www.linuxdevcenter.com/linux/cmd/

WEB_38, 2006. The GNU C Programming Tutorial, 30/06/2006.
http://www.crasseux.com/books/ctutorial/Concept-index.html

WEB_39, 2006. Prof. C. Edward Chow's web page - RTLinux, 30/06/2006.
http://piglet.uccs.edu/~chow/pub

WEB_40, 2006. PEER Strong Motion Database, 01/07/2006.
http://peer.berkeley.edu/smcat/search.html

WEB_41, 2006. Afet İşleri Genel Müd. Deprem Araştırma Dairesi - Türkiye Ulusal Kuvvetli Yer Hareketi Programı (TKYHP), 01/07/2006. http://angora.deprem.gov.tr/,
http://www.deprem.gov.tr

WEB_42, 2006. Tubitak Marmara Araştırma Grubu MAM, 01/07/2006.
http://www.mam.gov.tr/enstituler/ydbe/index.html

WEB_43, 2006. Ankara Sivil Savunma Deprem Simülasyon Merkezi , 03/07/2006.
http://www.ssgm.gov.tr/simulasyon.asp

WEB_44, 2006. Hyogo Earthquake Engineering Research Center, 03/07/2006.
http://www.bosai.go.jp/hyogo/ehyogo/index.html

WEB_45, 2006. University Consortium on Instructional Shake Tables UCIST, 03/07/2006.
http://cive.seas.wustl.edu/wusceel/ucist//

APPENDIX

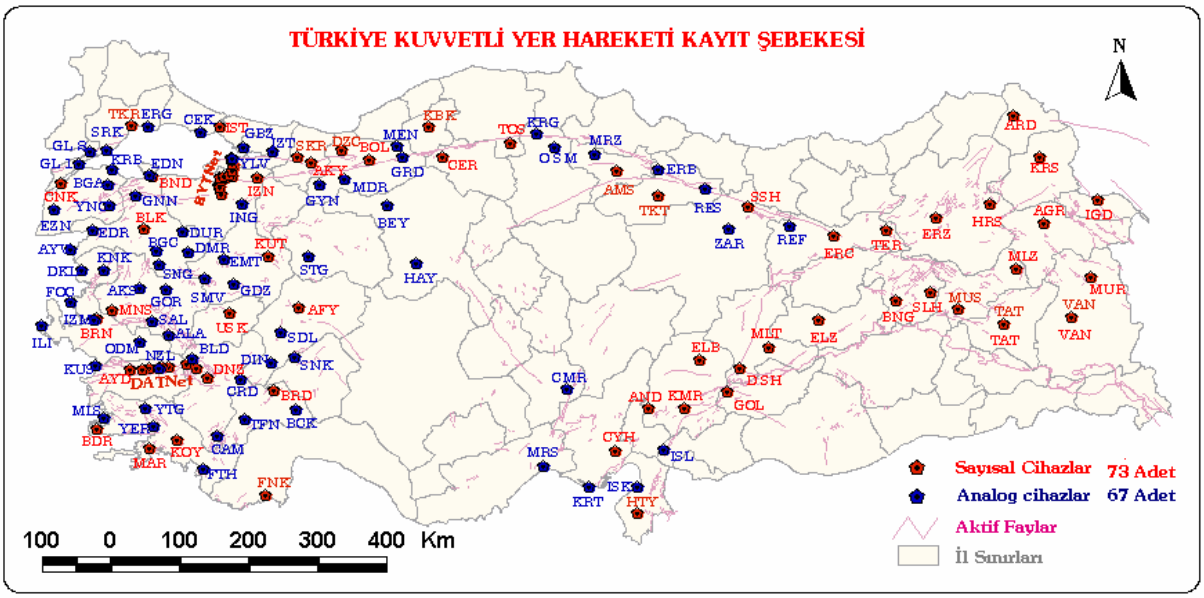


Figure A.1. Strong ground motion record network of Turkey (Source : ERD)

Figure A.2. Strong ground motion record network and local networks of Turkey (Source : ERD)

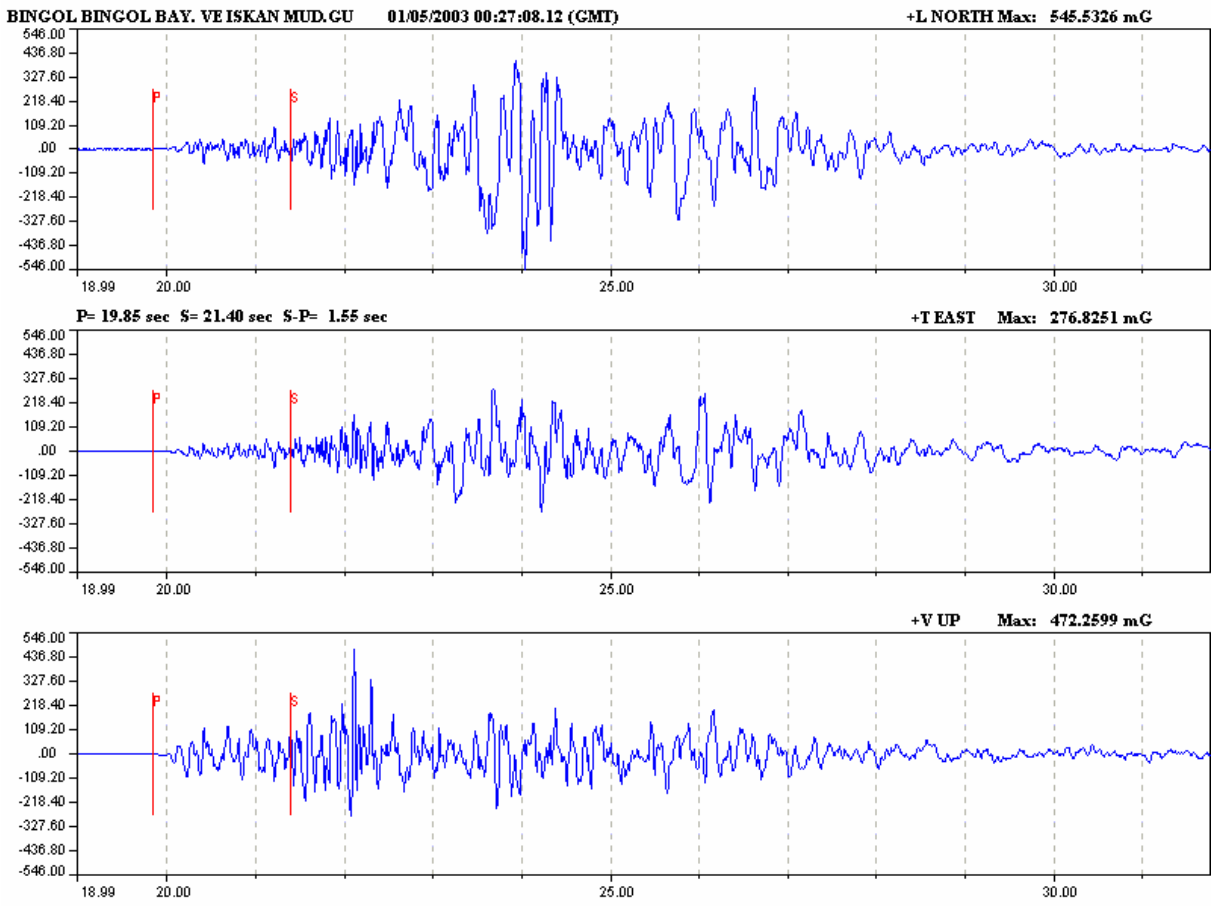


Figure A.3. 01 May 2003 Bingöl earthquake 00:27 (M=6.1) all components of maximum acceleration records (P and S waves are marked on the acceleograms) (Source : ERD)

BİNGÖL

Nüfusu : 232.273
İstasyon Kısaltması: BNG
Cihazın Kurulduğu Yer : Bayındırlık ve İskan Müdürlüğü
Cihazın Adı : GSR-16
Cihazın Koordinatları : 38.886N - 40.501E

Genel Jeolojisi : Bingöl Ovasının zemini Pleistosen yaşlı sıkışmış detritik depozitlerden oluşmuştur. Serinin alt seviyeleri, iri blok ve çakıllardan, üst seviyeleri ise çakıl, kum ve silt karışımından ibarettir. Üst seviyelerde de yer yer bloklara rastlanmaktadır. Bu seri bazı seviyelerde çok sıkışmış, bazı seviyelerde ise, gevşek dökülebilen nitelikte depozitler veya taraçalar halindedir. İçlerinde genellikle gnays, kuvars, kireçtaşı, şist, andezit, bazalt ve lav parçaları görülmektedir. Bu serinin kalınlığı 30-40 metredir(Tabban, 1980).

Deprem Durumu : Birinci derecede tehlikeli deprem bölgesindedir.
15.12.1934 M=5.8, 28.05.1940 M=5.2, 20.08.1966 M=5.5, 24.09.1968 M=5.1, 22.05.1971 M=6.7

Figure A.4. Bingöl station information from ERD

```
//STRONG GROUND MOTION RECORDS OF TURKiYE
//PLACE              :BINGOL BINGOL BAY. VE ISKAN MUD.GU
//RECORDER TYPE      :GSR16 (GeoSys)
//RECORDER SERIAL NO  :02299
//COORDINATES        :38.897N - 40.503E
//NATURAL FREQUENCY(Hz) :
//CRITICAL DAMPING    :
//TRIGGER DATE       :01/05/2003 00:27:08.12 (GMT)
//PRE-EVENT TIME(sec)  :  20
//TIME SYNC STATUS    :OK
//DIRECTIONS         :+L NORTH  +T EAST  +V UP
//NO. OF DATA        :  6474
//SAMPLE INTERVAL    :    .01000000
//MAX. VALUES(mG)    :(L)  545.5326 (T)  276.8251 (V)  472.2599
//EQ DATE            :2003.05.01 00:27:04
//EQ EPICENTER COORD.  :38.94N - 40.51E
//EQ MAGNITUDE       :6.1Md
//EQ DEPTH(km)       :6.0KM
//Copyright  EARTHQUAKE RESEARCH DEPARTMENT
//GENERAL DIRECTORATE OF DISASTER AFFAIRS
```

Figure A.5. Header of unprocessed acceleration data file of 01 May 2003 Bingöl earthquake (Source : ERD)

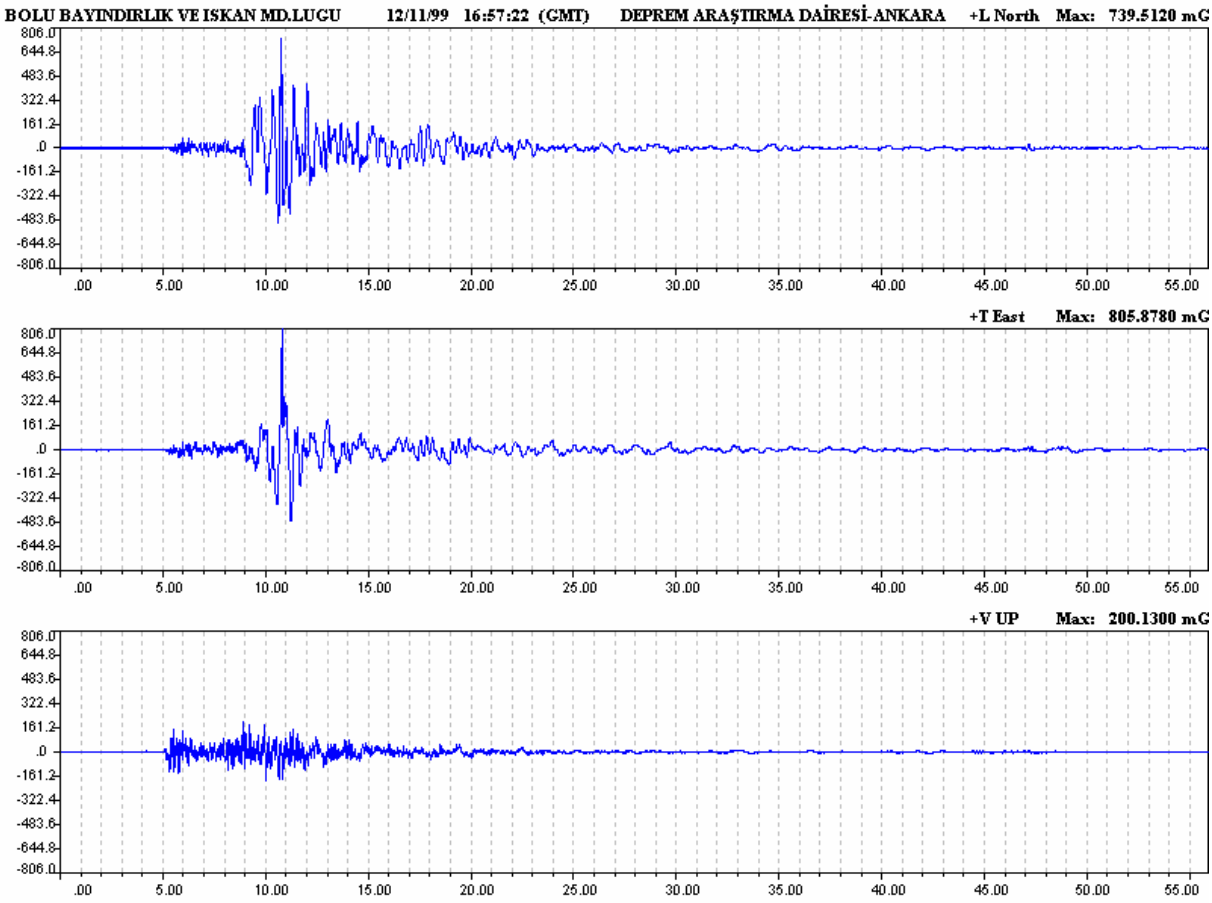


Figure A.6. 12 November 1999 Düzce earthquake 16:57 (Mw=7.2) all components of maximum acceleration records (Source: ERD)

BOLU

Nüfusu : 554.382
İstasyon Kısaltması: BOL
Cihazın Kurulduğu Yer : Bayındırlık ve İskan Müdürlüğü
Cihazın Adı : GSR-16
Cihazın Koordinatları : 40.747N - 31.610E

Genel Jeolojisi : Şehir, Pleistosen yaşlı eski alüvyona ait çakıl, kum, kumtaşı, marn ve konglomeralardan oluşmuş bir zemin üzerinde kurulmuştur. Tepe kısmını çevreleyen hafif meyilli yamaçlar, moloz ve toprak karışımından; ova kısmı ise kil, kum ve topraktan oluşmuş genç alüvyondan ibarettir. Yeraltı suyu yamaçlarda 3-4 m., ova bölgesinde ise 0.5-1 m. derinliktedir(Tabban, 1980).

Deprem Durumu : Birinci derecede tehlikeli deprem bölgesindedir.
24.11.1863 M=5.5, 08.02.1945 M=4.9, 19.05.1947 M=4.6, 26.05.1957 M=4.9, 26.05.1957 M=5.9, 27.05.1957 M=5.8, 17.06.1957 M=5.1

Figure A.7. Bolu station information from ERD

```
//STRONG GROUND MOTION RECORDS OF TURKiYE
//PLACE              :Bolu Bay. ve Isk. Mudurlugu
//RECORDER TYPE      :GSR18 (GeoSys)
//RECORDER SERIAL NO   :682
//COORDINATES        :40.747N - 31.610E
//NATURAL FREQUENCY(Hz) :
//CRITICAL DAMPING     :
//TRIGGER DATE       :12/11/1999 16:57:22    (GMT)
//PRE-EVENT TIME(sec)  :   5
//TIME SYNC STATUS     :OK
//DIRECTIONS         :+L NORTH  +T EAST  +V UP
//NO. OF DATA        :  5590
//SAMPLE INTERVAL    :     .01000000
//MAX. VALUES(mG)    :(L)   739.5120 (T)   805.8780 (V)   200.1300
//EQ DATE            :12/11/1999 16:57:20    (GMT)
//EQ EPICENTER COORD.  :40.74N - 31.21E
//EQ MAGNITUDE       :    7.2 Mw
//EQ DEPTH(km)       :    25.0
//Copyright   EARTHQUAKE RESEARCH DEPARTMENT
//GENERAL DIRECTORATE OF DISASTER AFFAIRS
```

Figure A.8. Header of unprocessed acceleration data file of 12 November 1999 Düzce earthquake (Source : ERD)

Date	Time (GMT)	N-S (mG)	E-W (mG)	V (mG)	S-P (sec)	Instrument	Epicenter	Station
12.11.1999	16:59:30	1.5	2.0	1.0	N/A	GSR-16	Duzce-Bolu	MNS
12.11.1999	16:59:28	3.6	3.5	1.7	N/A	GSR-16	Duzce-Bolu	DNZ
12.11.1999	16:59:06	3.9	3.3	1.1	N/A	GSR-16	Duzce-Bolu	CNK
12.11.1999	16:58:59	5.7	6.1	1.8	N/A	GSR-16	Duzce-Bolu	TKR
12.11.1999	16:58:49	2.4	2.2	0.9	60.23	GSR-16	Duzce-Bolu	AYD
12.11.1999	16:58:40	1.5	1.8	0.8	54.58	GSR-16	Duzce-Bolu	BRN
12.11.1999	16:58:21	3.1	3.1	1.4	42.5	GSR-16	Duzce-Bolu	USK
12.11.1999	16:58:17	2.7	2.4	1.7	40.40	GSR-16	Duzce-Bolu	BLK
12.11.1999	16:58:03	7.9	7.6	4.1	30.5	GSR-16	Duzce-Bolu	TOS
12.11.1999	16:58:01	8.0	10.0	3.5	28.08	GSR-16	Duzce-Bolu	AFY
12.11.1999	16:57:55	17.1	20.6	9.4	23.48	GSR-16	Duzce-Bolu	KUT
12.11.1999	16:57:54	9.3	8.0	4.8	25.63	GSR-16	Duzce-Bolu	BRS
12.11.1999	16:57:53	9.0	5.2	8.2	21.65	GSR-16	Duzce-Bolu	IST
12.11.1999	16:57:34	17.3	24.7	11.5	9.38	GSR-16	Duzce-Bolu	SKR
12.11.1999	**16:57:22**	**739.5**	**805.8**	**200.1**	**4.21**	**GSR-16**	**Duzce-Bolu**	**BOL**
12.11.1999	16:57:20	513.7	407.6	339.61	N/A	SMA-1	Duzce-Bolu	DZC
12.11.1999	16:57:20	22.2	23.8	22.4	N/A	SMA-1	Duzce-Bolu	IZT
12.11.1999	16:57:20	24.8	27.8	24.9	N/A	SMA-1	Duzce-Bolu	GYN
12.11.1999	16:57:20	22.0	21.4	9.8	N/A	SMA-1	Duzce-Bolu	IZN
12.11.1999	16:57:20	58.3	120.9	63.1	N/A	SMA-1	Duzce-Bolu	MDR

Figure A.9. 12 November 1999 Düzce earthquake 16:57 (Mw=7.2) main shock records

A.10. Scilab Function For Acausal Butterworth Filter

```
//Function name : Butter
//Usage :  function xdd_flt = Butter(xdd,order,fc)
//Acausal nth-order low-cut Butterworth filter

function xdd_flt = Butter(xdd,order,fc)

flt_butt = iir (order, 'hp', 'butt', [fc 0], [0 0]); //designs the filter
xdd_flt = flts (xdd' , flt_butt); //applies the filter to b_inp to obtain b_out.
xdd_flt = xdd_flt';

endfunction
```

A.11. Scilab Function For Fast Fourier Transformation (FFT)

```
//Function name : fft_func
//Usage :  [fft_mag , fft_freq] = fft_func(inp_data,dt)

//Fast Fourier Transformation FFT
//This function returns the magnitude and frequencies of the function inp_data.

function [fft_mag , f] = fft_func(inp_data,dt)

out_data = fft(inp_data , -1); //-1  argument refers to the sign of the exponent,not to "inverse"
N = length(out_data);
//the fft response is symetric we retain only the first N/2 points
fft_mag = abs(out_data(1:(int(N/2)+1))); //Magnitude
f = ((0:int(N/2))/(2*int(N/2)*dt))'; //Frequency in Hz

endfunction
```

A.12. Numerical Integration Scheme

Once *int_u* system is defined, then integration of any signal can be performed. For defining *int_u* system, some parameters are utilized:

syslin defines a linear system,
A,B,C, and *D* matrices are matrices of the state-space representation,
x0 is initial state vector, and
'c' is used for a continuous time system.

csim simulates time response of linear systems.

```
//Function name : Numerical_Integ
//Usage :  function [xd,x] = Numerical_Integ(xdd)

//Numerical integration
function [xd,x] = Numerical_Integ(xdd)

A=[0 1; 0 0]; B=[0;1]; C=[0 1]; D=0;
x0=[0 0]';//initial state
int_u = syslin('c',A,B,C,D,x0);
xd = csim( xdd, t, int_u );

A=[0 1; 0 0]; B=[0;1]; C=[0 1]; D=0;
x0=[0 0]';//initial state
int_u = syslin('c',A,B,C,D,x0);
x = csim( xd, t, int_u );

endfunction
```

A.13. Scilab Code For 01 May 2003 Bingöl Earthquake

```
//EQ data processing
//Author: Gokce Kinay

//hi indicates helper variable
clear; //delete all variables
clc(); //clear command window
delete all; //delete all the graphics objects of the figure //clf(); clear the current graphic
window, clear figure

stacksize(5e6); //default stacksize may be 5e6 byte(5 MB), here it is increased to 50 MB
DIR = pwd(); //current directory
chdir(DIR);

// FUNCTIONS
//************
//Function name : Numerical_Integ
//Usage :  function [xd,x] = Numerical_Integ(xdd,#data)
function [xd,x] = Numerical_Integ(xdd)
…
endfunction

//Function name : Butter
//Usage :  function xdd_flt = Butter(xdd,order,fc)
function xdd_flt = Butter(xdd,order,fc)
…
endfunction

//Function name : fft_func
//Usage :  [fft_mag , fft_freq] = fft_func(inp_data,dt)
function [fft_mag , f] = fft_func(inp_data,dt)
…
endfunction
//************
// END OF FUNCTIONS


//DATA
//******
//ANGORA
//01 MAY 2003 BINGOL EQ ( 00:27  Station:Bingol)
h1 =
fscanfMat(DIR+'/Desktop/GOK/tubitakprj/dataprocessing/angora/MAYIS_03_BINGOL_max
/200305010027A-BNG.txt');
xdd_uncor = h1(:,1);//NORTH-SOUTH component
```

```
//xdd_uncor = h1(:,2);//EAST-WEST component
//xdd_uncor = h1(:,3);//UP component


clear h1; #_of_original_data = length(xdd_uncor)
dt = 0.01;  //time interval
//******

#data = length(xdd_uncor); #data
t = 0:dt:dt*(#data-1);

[mag_xdd_uncor , f_xdd_uncor] = fft_func(xdd_uncor,dt); //for decision of fc
[xd_uncor,x_uncor] = Numerical_Integ(xdd_uncor);


//STEP1 Mean-correction of uncorrected acceleration data
//************************************************
//pre-event signal is 20 sec.
pre_event_step = 20 / dt;
xdd_meancor = xdd_uncor - mean(xdd_uncor(1:pre_event_step));
[mag_xdd_meancor , f_xdd_meancor] = fft_func(xdd_meancor,dt);


//STEP2 Numerical integration of the mean-corrected acceleration to velocity
//************************************************************
 [xd_meancor,x_meancor] = Numerical_Integ(xdd_meancor);

p=2; zer = zeros(length(t),1);
subplot(p,1,1); plot2d(t,[xd_meancor],[3]); xtitle(['For decision of the processing
procedure'],'time(sec)','xd_meancor(cm/sec)');
subplot(p,1,2); plot2d(t,[x_meancor],[6]); xtitle(['x_meancor'],'time(sec)','x_meancor(cm)')


//STEP3 fit a first-order polynomial to the velocity, v = c1 + c2*t
//*****************************************************
//Linear fitting with 'datafit'
Z = [t'; xd_meancor];
deff('[e]=G(a,z)','e=z(2)-a(1)-a(2)*z(1)-a(3)*z(1)^2')
a0 = [1;1;1];
[c,er] = datafit(G,Z,a0);//the command datafit is improved version of fit_dat

//deff('[v]=h6(t)','v=c(1)+c(2)*t+c(3)*t^2') //in case of quadratic fit
deff('[v]=h6(t)','v=c(1)+c(2)*t')
//xd_curvefitted = h6(t)';
//plot(t,xd_curvefitted,'-',t,xd_meancor,'+')
//xtitle('Quadratic fitting and original data','t','xd')
clear h6;
//STEP4 Baseline-correction of acceleration
```

```
//*********************************
//Baseline correction provides flatter baselines and averages the baseline to zero.

//This improves the accuracy of integrals, the appearance of the spectrum, and the quality of a result from subtracting one spectrum from another.

//deff('[der_v]=p(t)','der_v=c(2)+2*c(3)*t') //derivative of curve fitted velocity der_v = c2 + 2*c3*t  //in case of quadratic fit
deff('[der_v]=p(t)','der_v=c(2)') //derivative of curve fitted velocity der_v = c2 + 2*c3*t
xdd_basecor = xdd_meancor - p(t)';


//PREPERATIONS FOR LOW-CUT BUTTERWORTH FILTER
//*****************************************************
//ZERO-PADDING
order = 1; fc = 0.05; //corner frequency of fc [Hz]
nroll = order/4;
T_pad = 1.5*nroll/fc; //total length of zeros to be added to the record
#zeros = T_pad / dt; //zeros added as pre- and the other of zeros added as post-

new_size = 2 * #zeros + length(xdd_basecor);
//assembly in the new data vector
xdd_zerop = zeros(new_size,1); //zero padded xdd
xdd_zerop((#zeros+1) : (#zeros+length(xdd_basecor))) = xdd_basecor;
new_size
t_final = 0:dt:dt*(length(xdd_zerop)-1);


//FILTERING
//***********
xdd_flt = Butter(xdd_zerop,order,fc);
xdd_final = xdd_flt;


//STEP Numerical integration of the filtered acceleration to velocity & displacement
//*****************************************************************
 [xd_final,x_final] = Numerical_Integ(xdd_final);


//STEP7  FFT - Computation of FAS of  baseline-corrected and filtered acceleration
//*****************************************************************
 [mag_xdd_final , f_xdd_final] = fft_func(xdd_final((#zeros+1) :
(#zeros+length(xdd_basecor))),dt);
```

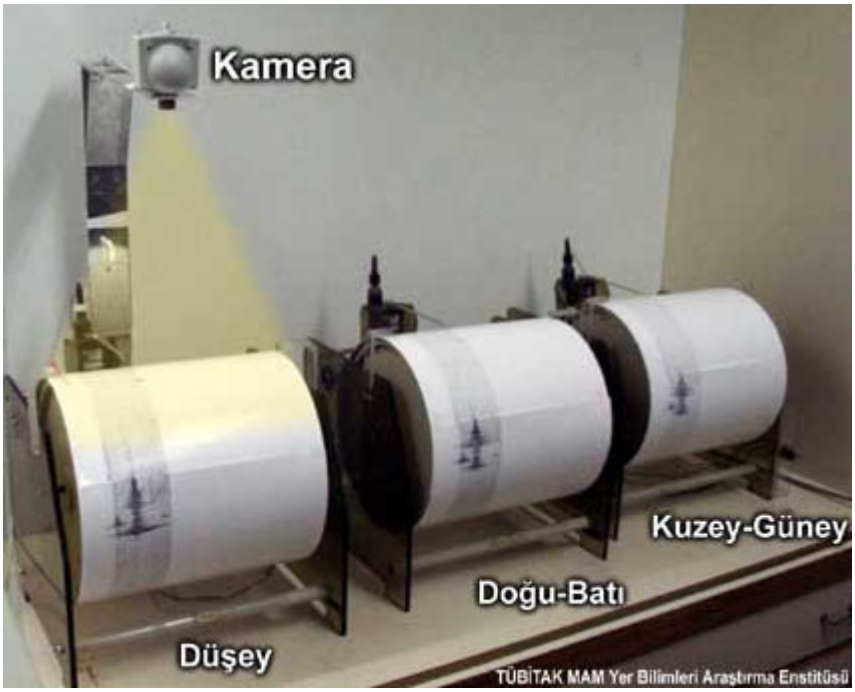


Figure A.14. Webcam view

Figure A.15. View of a seismogram

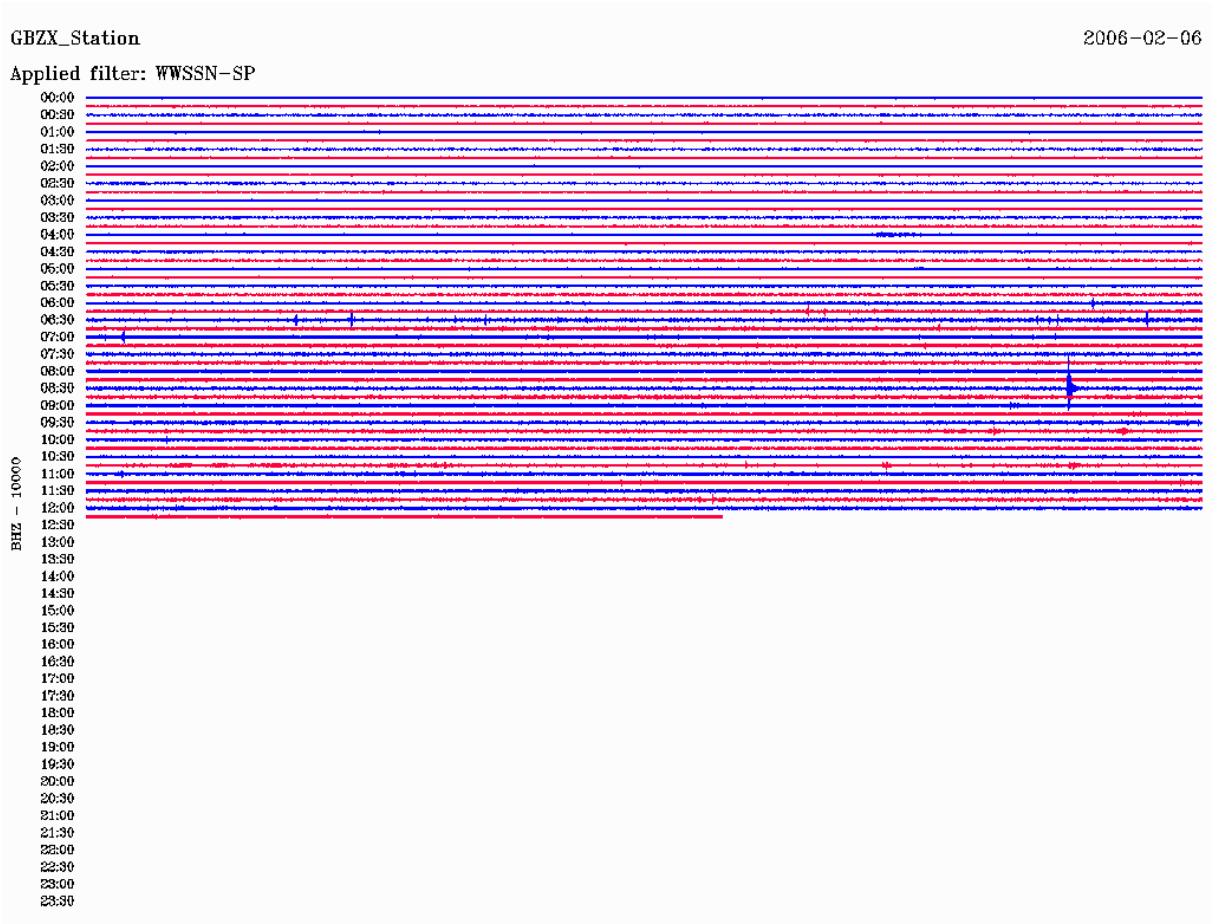


Figure A.16. View of a recording digital seismogram from MAM

A.17. ZeroPosiCalib.c

```
/*************************************************************************/
/*                          Calibration of Table's Zero-Position         */
/*  This example shows how to use the powerdaq API to perform a single update  */
/*  The update is performed in a software timed fashion                 */
/*  that is appropriate for slow speed generation (up to 500Hz).        */
/*  Copyright (C) 2001 United Electronic Industries, Inc.               */
/*  All rights reserved.                                                */
/*************************************************************************/
/* Updated & arranged by Gokce KINAY & Gursoy TURAN                     */
/* Last Updated : June 2006 IYTE Izmir                                  */
/*************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <sys/types.h>
#include <unistd.h>
#include <math.h>
#include "win_sdk_types.h"
#include "powerdaq.h"
#include "powerdaq32.h"
#include "ParseParams.h"

typedef enum _state
{
  closed,
  unconfigured,
  configured,
  running
} tState;

typedef struct _singleAoData
{
  int board;                  // board number to be used for the AI operation
  int handle;                 // board handle
  int abort;
  int adapterType;            // adapter type, should be atMF or atPD2AO
  unsigned long channelList[64];
  int nbOfChannels;           // number of channels
  int nbOfPointsPerChannel;   // number of samples per channel
  double updateRate;          // sampling frequency on each channel
  tState state;               // state of the acquisition session
} tSingleAoData;
int InitSingleAO(tSingleAoData *pAoData);
int SingleAO(tSingleAoData *pAoData, double *bufferAO);
void CleanUpSingleAO(tSingleAoData *pAoData);
void SigInt(int signum);
static tSingleAoData G_AoData;
```

```
void SingleAOExitHandler(int status, void *arg)
{
  CleanUpSingleAO((tSingleAoData *)arg);
}
int InitSingleAO(tSingleAoData *pAoData)
{
  ……..
  return 0;
}
int SingleAO(tSingleAoData *pAoData, double *bufferAO)
{
  ……..
  value = (bufferAO[0] + 10.0) / 20.0 * 0xFFF;
  //division by 20 comes from the total voltage interval +- 10 V
  retValAO = _PdAOutPutValue(pAoData->handle, value);//output is sent here
  ……..
  return retValAO;
}
void CleanUpSingleAO(tSingleAoData *pAoData)
{
 ……..
}
void SigInt(int signum)
{
 ……..
}

int main(int argc, char *argv[])
{
  double *bufferAO = NULL;
  int i, m, n, n_of_samples = 1;
  double ZeroPosiVoltage, ZeroPosiVoltageRev;

  PD_PARAMS params = {0, 1, {0}, 1000.0, 0, n_of_samples};

  ParseParameters(argc, argv, &params);

  // AO initializes acquisition session parameters
  G_AoData.board = params.board;
  G_AoData.nbOfChannels = params.numChannels;
  for(i=0; i<params.numChannels; i++)
    G_AoData.channelList[i] = params.channels[i];
  G_AoData.handle = 0;
  G_AoData.abort = FALSE;
  G_AoData.nbOfPointsPerChannel = params.numSamplesPerChannel;
  G_AoData.updateRate = params.frequency;
  G_AoData.state = closed;

  printf("\n"); printf("This is an PDL-MFx board\n"); printf("\n");
```

```
  printf("Calibration of Table's Zero-position \n"); printf("\n");


  // setup exit handler that will clean-up the acquisition session
  // if an error occurs
  on_exit(SingleAOExitHandler, &G_AoData);

  signal(SIGINT, SigInt);

  // allocate memory for the generation bufferAO
  bufferAO = (double *) malloc(G_AoData.nbOfChannels *
G_AoData.nbOfPointsPerChannel *
                        sizeof(double));

  if(bufferAO == NULL)
  {
    printf("SingleAO: could not allocate enough memory for the generation bufferAO\n");
    exit(EXIT_FAILURE);
  }

  // initializes acquisition session
  InitSingleAO(&G_AoData);

  ZeroPosiVoltage = 2;
  //2 Volts , table goes through CW switch (left)

  ZeroPosiVoltageRev = -2;
  //-2 Volts (in reverse direction), table goes through CCW switch (right)

  m=0;//moving the table to + or - max limits of table
  while(m<308000)//for time arrangement
  {
     SingleAO(&G_AoData, &ZeroPosiVoltage); m+=1;
     //printf("HELLO\n");
  }

  n=0;//moving the table to zero-position
  while(n<106000)//for time arrangement
  {
     SingleAO(&G_AoData, &ZeroPosiVoltageRev); n+=1;
  }

  // Cleanup acquisition
  CleanUpSingleAO(&G_AoData);

  return 0;
}
```

A.18. SingleAIO.c

```
/*                        Single update analog output example                  */
/*                                                                             */
/*  This example shows how to use the powerdaq API to perform a single update  */
/*  The update is performed in a software timed fashion                        */
/*  that is appropriate for slow speed generation (up to 500Hz).               */
/*                                                                             */
/*                        Single scan analog input example                     */
/*                                                                             */
/*  This example shows how to use the powerdaq API to perform a single scan    */
/*  acquisition. The acquisition is performed in a software timed fashion      */
/*  that is appropriate for slow speed acquisition (up to 500Hz).              */
/*                                                                             */
/*     Copyright (C) 2001 United Electronic Industries, Inc.                   */
/*     All rights reserved.                                                    */
/*******************************************************************************/
/* Updated & arranged by Gokce KINAY & Gursoy TURAN                            */
/* Last Updated : June 2006 IYTE Izmir                                         */
/*******************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <sys/types.h>
#include <unistd.h>
#include <math.h>
#include "win_sdk_types.h"
#include "powerdaq.h"
#include "powerdaq32.h"

#include "ParseParams.h"

// Macros
#define max(a,b) ((a)>(b) ? (a):(b))
#define abs(a) ((a)<0 ? -(a):(a))

// Global variables
int siz = 5590; //size of eq data is given here
int sizsiz = 6000; //size of arrays & last # in params

typedef enum _state
{
  closed,
  unconfigured,
  configured,
  running
} tState;
```

```
typedef struct _singleAoData
{
  int board;                  // board number to be used for the AI operation
  int handle;                 // board handle
  int abort;
  int adapterType;            // adapter type, should be atMF or atPD2AO
  unsigned long channelList[64];
  int nbOfChannels;           // number of channels
  int nbOfPointsPerChannel;   // number of samples per channel
  double updateRate;          // sampling frequency on each channel
  tState state;               // state of the acquisition session
} tSingleAoData;

typedef struct _singleAiData
{
  int board;                  // board number to be used for the AI operation
  int handle;                 // board handle
  int nbOfChannels;           // number of channels
  int nbOfSamplesPerChannel;  // number of samples per channel
  unsigned long channelList[64];
  double scanRate;            // sampling frequency on each channel
  int polarity;               // polarity of the signal to acquire, possible value
                              // is AIN_UNIPOLAR or AIN_BIPOLAR
  int range;                  // range of the signal to acquire, possible value
                              // is AIN_RANGE_5V or AIN_RANGE_10V
  int inputMode;              // input mode possible value is AIN_SINGLE_ENDED or
AIN_DIFFERENTIAL
  int trigger;
  tState state;               // state of the acquisition session
} tSingleAiData;

//AO
int InitSingleAO(tSingleAoData *pAoData);
int SingleAO(tSingleAoData *pAoData, double *bufferAO);
void CleanUpSingleAO(tSingleAoData *pAoData);
void SigInt(int signum);

static tSingleAoData G_AoData;

//AI
int InitSingleAI(tSingleAiData *pAiData);
int SingleAI(tSingleAiData *pAiData, double *bufferAI);
void CleanUpSingleAI(tSingleAiData *pAiData);

static tSingleAiData G_AiData;

// AO exit handler
void SingleAOExitHandler(int status, void *arg)
{
  CleanUpSingleAO((tSingleAoData *)arg);
```

```
}

// AI exit handler
void SingleAIExitHandler(int status, void *arg)
{
  CleanUpSingleAI((tSingleAiData *)arg);
}

int InitSingleAO(tSingleAoData *pAoData)
{
  ……..
  return 0;
}

int InitSingleAI(tSingleAiData *pAiData)
{
  ……..
  return 0;
}

int SingleAO(tSingleAoData *pAoData, double *bufferAO)
{
  int retValAO;
  int  i;

  DWORD aoCfg;
  DWORD value;

  printf("\n");
  printf("NEW EARTHQUAKE DATA\n");
  printf("******Function SingleAO******\n");

  // set configuration - _PdAOutReset is ...called inside _PdAOutSetCfg
  aoCfg = 0;
  retValAO = _PdAOutSetCfg(pAoData->handle, aoCfg, 0);

  if(retValAO < 0)
  {
    printf("SingleAO: _PdAOutSetCfg failed with code %x\n", retValAO);
    exit(EXIT_FAILURE);
  }

  pAoData->state = configured;

  retValAO = _PdAOutEnableConv(pAoData->handle, TRUE);
  if (retValAO < 0)
  {
    printf("SingleAO: _PdAOutEnableConv error %d\n", retValAO);
    exit(EXIT_FAILURE);
  }
```

```
  // Start SW trigger
  retValAO = _PdAOutSwStartTrig(pAoData->handle);
  if (retValAO < 0)
  {
    printf("SingleAO: PdAOutSwStartTrig failed with %d.\n", retValAO);
    exit(EXIT_FAILURE);
  }

  pAoData->state = running;

  value = (bufferAO[0] + 10.0) / 20.0 * 0xFFF;//division by 20 comes from the total voltage
interval +- 10 V

  // write update value to the board
  i=0;
  while(i<450)//dt=0.01sec; 450 is selected for time arrangement
            //dt=0.005sec; 180 is selected for time arrangement
  {
   retValAO = _PdAOutPutValue(pAoData->handle, value);//output is sent here
   i+=1;
  }
  printf("voltageAO: %f   value: %d   retValAO: %d\n",*bufferAO, value, retValAO);

  if(retValAO < 0)
  {
   printf("SingleAO: _PdAOutPutValue failed with %d.\n", retValAO);
   exit(EXIT_FAILURE);
  }

  fflush(stdout);
  if(pAoData->abort)

  usleep(1.0E6/pAoData->updateRate);

  return retValAO;
}

int SingleAI(tSingleAiData *pAiData, double *bufferAI)
{
  ……..
  return retValAI;
}

void CleanUpSingleAO(tSingleAoData *pAoData)
{
  ……..
}

void CleanUpSingleAI(tSingleAiData *pAiData)
```

```
{
  ……..
}

void SigInt(int signum)
{
  ……..
}

int main(int argc, char *argv[])
{
  FILE *eq_xg; FILE *eq_xddg;
  FILE *volt_data;
  FILE *read_accel_data; FILE *scaled_disp; FILE *scaled_velo; FILE *scaled_accel;

  double EqVoltWeight, DispScale;
  double *bufferAO = NULL, *bufferAI = NULL, voltageAO[sizsiz], abs_voltageAO[sizsiz];
  double max_voltageAO, read_accel[sizsiz];
  int i,j,k,index_of_max_voltageAO;
  int nbOfBoards, gain = 0;
  float xg[sizsiz],  *ptr_xg, abs_maxEqDisp, abs_xg[sizsiz], sca_abs_maxDisp;
  float xdg[sizsiz],  *ptr_xdg, abs_maxEqVelo, abs_xdg[sizsiz], sca_abs_maxVelo;
  float xddg[sizsiz],  *ptr_xddg, abs_maxEqAccel, abs_xddg[sizsiz], sca_abs_maxAccel;
  int n_of_samples=1;// IMPORTANT

  PD_PARAMS params = {0, 1, {0}, 1000.0, 0, n_of_samples};

  ParseParameters(argc, argv, &params);

  // AO initializes acquisition session parameters
  G_AoData.board = params.board;
  G_AoData.nbOfChannels = params.numChannels;
  for(i=0; i<params.numChannels; i++)
     G_AoData.channelList[i] = params.channels[i];
  G_AoData.handle = 0;
  G_AoData.abort = FALSE;
  G_AoData.nbOfPointsPerChannel = params.numSamplesPerChannel;
  G_AoData.updateRate = params.frequency;
  G_AoData.state = closed;

  PD_PARAMS paramsAI = {0, 1, {0}, 1000.0, 0, n_of_samples};
  // AI initializes acquisition session parameters
  G_AiData.board = paramsAI.board;
  G_AiData.nbOfChannels = paramsAI.numChannels;
  for(i=0; i<paramsAI.numChannels; i++)
     G_AiData.channelList[i] = paramsAI.channels[i] | (gain << 6);
  G_AiData.handle = 0;
  G_AiData.nbOfSamplesPerChannel = paramsAI.numSamplesPerChannel;
  G_AiData.scanRate = paramsAI.frequency;
```

```
  G_AiData.polarity = AIN_BIPOLAR;
  G_AiData.range = AIN_RANGE_10V;
  G_AiData.inputMode = AIN_DIFFERENTIAL;
  G_AiData.state = closed;
  if(paramsAI.trigger == 1)
    G_AiData.trigger = AIB_STARTTRIG0;
  else if(paramsAI.trigger == 2)
    G_AiData.trigger = AIB_STARTTRIG0 + AIB_STARTTRIG1;
  else
    G_AiData.trigger = 0;

  printf("This is an PDL-MFx board\n"); printf("\n");

  //Earthquake velocity data is given manually from keyboard
  ptr_xdg = &xdg[0];
  for (k=0;k<=(siz-1);k++)
  {
     scanf("%f", &xdg[k]);
  }

   //Earthquake displacement and acceleration data are read from the current directory
  eq_xg = fopen("BOL_UP_disp_vec.txt" , "r");  eq_xddg = fopen("BOL_UP_accel_vec.txt" ,
"r");
  ptr_xg = &xg[0];  ptr_xddg = &xddg[0];
  for (k=0;k<=(siz-1);k++)
  {
     fscanf(eq_xg , "%f", &xg[k]);
     fscanf(eq_xddg , "%f", &xddg[k]);
  }
  fclose (eq_xg); fclose (eq_xddg);

  for(i=0; i<=(siz-1); i++)
  {
    abs_xg[i] = abs(xg[i]);
    abs_maxEqDisp = max(abs_maxEqDisp, abs_xg[i]);

    abs_xdg[i] = abs(xdg[i]); //absolute value of Eq velocity data
    abs_maxEqVelo = max(abs_maxEqVelo, abs_xdg[i]); //max abs Eq velocity data

    abs_xddg[i] = abs(xddg[i]);
    abs_maxEqAccel = max(abs_maxEqAccel, abs_xddg[i]);
  }

  //Check if max of earhquake displacement data is in the allowable displacement range of
table +-7 cm.
  for(i=0; i<=(siz-1); i++)
  {
    if (abs_maxEqDisp > 7) //ATTENTION : UNITS ARE cm & sec
    {
       DispScale = 7./abs_maxEqDisp;
```

```
        xg[i] = DispScale * xg[i]; //xg is scaled
        xdg[i] = DispScale * xdg[i]; //xdg is scaled
        xddg[i] = DispScale * xddg[i]; //xddg is scaled
      }
    }
    //max of scaled values
    for(i=0; i<=(siz-1); i++)
    {
    abs_xg[i] = abs(xg[i]);
    sca_abs_maxDisp = max(sca_abs_maxDisp, abs_xg[i]);

    abs_xdg[i] = abs(xdg[i]);
    sca_abs_maxVelo = max(sca_abs_maxVelo, abs_xdg[i]);

    abs_xddg[i] = abs(xddg[i]);
    sca_abs_maxAccel = max(sca_abs_maxAccel, abs_xddg[i]);
    }

    //print of max values
    printf("\n");
    printf("MAXIMUMS OF ABSOLUTE VALUES (unscaled)\n");
    printf("Maximum Absolute Ground Displacement : %f cm\n",abs_maxEqDisp);
    printf("Maximum Absolute Ground Velocity : %f cm/s\n",abs_maxEqVelo);
    printf("Maximum Absolute Ground Acceleration : %f cm/s^2\n",abs_maxEqAccel);
    printf("\n");

    printf("ATTENTION\n");
    printf("Table's allowable displacement limits are exceeded with this data\n");
    printf("Therefore displacement, velocity and acceleration data ");
    printf("are scaled by a SCALE FACTOR of %f\n",DispScale);
    printf("Same scale is used for displacement, velocity and acceleration data\n");
    printf("Because time scale is accepted as 1\n");

    printf("\n");
    printf("MAXIMUMS OF ABSOLUTE VALUES (scaled)\n");
    printf("Maximum Absolute Ground Displacement (scaled): %f cm\n",sca_abs_maxDisp);
    printf("Maximum Absolute Ground Velocity (scaled): %f cm/s\n",sca_abs_maxVelo);
    printf("Maximum Absolute Ground Acceleration (scaled): %f
cm/s^2\n",sca_abs_maxAccel);
    printf("\n");

    //eq data is converted into voltageAO (analog outputs have a fixed output range of +-10 V )
    EqVoltWeight = 10./25;
    printf("\n");
    printf("Scaled velocities are converted into voltage by a factor of EQVOLTWEİGHT :
%f\n",EqVoltWeight);
    printf("\n");

    for (j=0; j<=(siz-1); j++)
    {
```

```
    voltageAO[j] = xdg[j] * EqVoltWeight;
  }

  // setup exit handler that will clean-up the acquisition session
  // if an error occurs
  on_exit(SingleAOExitHandler, &G_AoData);

  signal(SIGINT, SigInt);

  // allocate memory for the generation bufferAO
  bufferAO = (double *) malloc(G_AoData.nbOfChannels *
G_AoData.nbOfPointsPerChannel *
                        sizeof(double));

  if(bufferAO == NULL)
  {
    printf("SingleAO: could not allocate enough memory for the generation bufferAO\n");
    exit(EXIT_FAILURE);
  }

  // allocate memory for the bufferAI
  bufferAI = (double *) malloc(G_AiData.nbOfChannels *
G_AiData.nbOfSamplesPerChannel *
                        sizeof(double));
  if(bufferAI == NULL)
  {
    printf("SingleAI: could voltageAO[sizsiz],not allocate enough memory for the acquisition
bufferAI\n");
    exit(EXIT_FAILURE);
  }

  // AI setup exit handler that will clean-up the acquisition session
  // if an error occurs
  on_exit(SingleAIExitHandler, &G_AiData);
  nbOfBoards = PdGetNumberAdapters();

  // initializes acquisition session
  InitSingleAO(&G_AoData);

  // initializes acquisition session
  InitSingleAI(&G_AiData);

  for(i=0; i<=(siz-1); i++)
  {
     //run the acquisition
     SingleAO(&G_AoData, &voltageAO[i]);

     //run the acquisition
     SingleAI(&G_AiData, bufferAI);
  }
```

```
  // Cleanup acquisition
  CleanUpSingleAO(&G_AoData);

  // Cleanup acquisition
  CleanUpSingleAI(&G_AiData);

  // free acquisition bufferAI
  free(bufferAI);

  //absolute value of voltageAO
  for(i=0; i<=(siz-1); i++)
  {
    abs_voltageAO[i] = abs(voltageAO[i]);
    //printf("voltageAO : %f\n",voltageAO[i]);

    //max voltage
    max_voltageAO = max(max_voltageAO, abs_voltageAO[i]);

    //index of maximum voltageAO
    if (abs_voltageAO[i] == max_voltageAO)
    {
      index_of_max_voltageAO = i;
    }
  }

  index_of_max_voltageAO = index_of_max_voltageAO + 1; //index started from 0

  for(i=0; i<=(siz-1); i++)
  {
    //for negative maximum values
    if (voltageAO[index_of_max_voltageAO-1] < 0)
    {
      max_voltageAO = voltageAO[index_of_max_voltageAO-1];
    }
  }
  printf("\n");
  printf("END OF DATA\n");
  printf("index of maximum voltageAO = %d\n",index_of_max_voltageAO);
  printf("!!!!!!MAXIMUM VOLTAGE!!!!!! : %f\n",max_voltageAO);
  printf("\n");

  return 0;

}
```

A.19. Some Explanations About SingleAIO.c

A C code contains *functions* and *variables*. The functions specify the tasks to be performed by the program. The *main* function, which is the brain of the code, establishes the overall logic of the code. It calls different functions to perform the necessary sub-tasks. All self standing C programs must have a *main* function.

All variables in C must be explicitly defined before use. The header files (*.h) which contain definitions of variables and functions are necessary for functioning of a program. They can be included as part of the standard C libraries or user can write his/her own code. The statement *#include <stdio.h>* tells the C compiler to insert the contents of the specified file and the compiler looks for the file in certain standard system directories due to the presence of < > notation. Header files in SingleAIO.c are explained roughly in Fig.4.7.

stdio.h	standart input/output library that defines I/O routines
stdlib.h	defines number conversion, storage allocation and similar tasks
math.h	defines mathematical routines
stdint.h	declares integer types
sys/types.h	defines data types
unistd.h	defines standard symbolic constants and types
win_sdk_types.h	contains data type definitions needed by the files of kernel driver and os shared library
powerdaq.h	contains driver constants and definitions for C/C++
powerdaq32.h	API function prototypes and structures for C/C++

Header files in SingleAIO.c

Number of data is given as global variables at the beginning of C file. The global variable *sizsiz* is an arbitrary integer greater than *siz*, but it shouldn't be too big. Because it indicates the size of related arrays in C code.

```
// Global variables
int siz = 5590; //size of eq data is given here
int sizsiz = 6000; //size of arrays & last # in params
```

Starting from the *main* function, names of links to text files are given by *FILE *name_of_link_to_the_file*. The dereferencing operator * indicates to the content of the memory allocation referenced by this link. These links are to displacement and acceleration data files or to the files which converted voltage data, scaled data and read acceleration data are written.

Local variables which are only defined in the relevent function are given as integer, double or floating-point standart C variables by *int*, *double* and *float*. Size of the earthquake array is given by *float xg[sizsiz]*. Number of samples which will be sent as output is determined as 1 by *int n_of_samples=1;* This is due to the selected programing style and will be explained in the following lines.

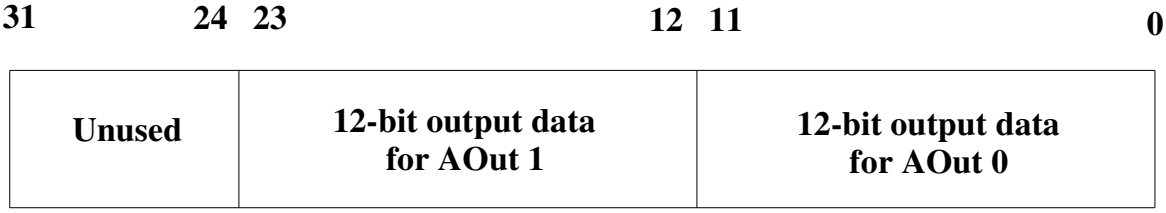
A certain amount of memory should be allocated for data acquisition by *malloc*, and should let it free after the acquisition by *free*. It is necessary to initialize the acquisition session of card before the operation, and then clean up it after the operation.

```
bufferAI = (double *) malloc(G_AiData.nbOfChannels * G_AiData.nbOfSamplesPerChannel
* sizeof(double));
……
free(bufferAI);
```

After all these preparations, voltageAO is ready to send to servo motor driver from analog output channel of PowerDAQ by means of function *SingleAO*. Number of samples to be sent as output is determined as 1. The project group have prefered to go to functions *SingleAO* and *SingleAI* once for every element of a loop. This loop is running from 0 to (earthquake data size-1) with a total size of earthquake data. The functions *SingleAO* and *SingleAI* sends voltage to output channels of PowerDAQ board and reads voltage from input channels of it in a single analog update and single analog scan fashion, respectively. If the *main* function is assumed as heart of a C code, the following part can be assumed as brain of Single AIO.c 's *main* function.

```
InitSingleAO(&G_AoData);
InitSingleAI(&G_AiData);
for(i=0; i<=(siz-1); i++)
{
        SingleAO(&G_AoData, &voltageAO[i]);
        SingleAI(&G_AiData, bufferAI);
}
CleanUpSingleAO(&G_AoData);
CleanUpSingleAI(&G_AiData);
```

The parts of C file created by United Electronic Industries, Inc. are mentioned roughly just to give a impression. The heart of functions *SingleAO* and *SingleAI* are given in the following lines. The single update method utilizes an application programming interface command from the program to write the digital form of the analog output value directly to a D/A converter.There exists a fixed channel list of Powerdaq board for analog output and it always contains channel 0 and 1 which are updated simultaneously. The analog output data format of board is given in Fig.4.10.



Analog output data format of PowerDAQ

The analog outputs have a fixed output range of ±10 Volts with a total voltage interval of 20 Volts. The following formula is utilized to convert voltage into binary code:
HexValue = ((Voltage + 10 V) / 20) * 0xFFF

```
value = (bufferAO[0] + 10.0) / 20.0 * 0xFFF;
retValAO = _PdAOutPutValue(pAoData->handle, value);//output is sent here
……
//input channel is read here
retValAI = _PdAInGetSamples(pAiData->handle, pAiData->nbOfChannels *
                    pAiData->nbOfSamplesPerChannel, rawBuffer, &numScans);
```

```
#!/bin/sh

insmod /usr/src/rtlinux/modules/rtl.o
insmod  /usr/src/rtlinux/modules/rtl_time.o
if [ -f /usr/src/rtlinux/modules/rtl_posixio.o ]; then
        insmod /usr/src/rtlinux/modules/rtl_posixio.o
fi
insmod /usr/src/rtlinux/modules/rtl_fifo.o
insmod /usr/src/rtlinux/modules/rtl_sched.o
if [ -f /usr/src/rtlinux/modules/psc.o ]; then
        insmod /usr/src/rtlinux/modules/psc.o
fi
exit 0
```

file-insmod.sh

```
 #!/bin/bash

rmmod pwrdaq
rmmod rtl_fifo
rmmod rtl_posixio
rmmod rtl_sched
rmmod rtl_time
rmmod rtl
```

rmmod-file.sh

A.20. Some Explanations About RTLsimpleAO.c

The number of earthquake velocity data is given by *aoData.nbOfPointsPerChannel = 5500.* The RTLinux function *pthread_make_periodic_np* marks a realtime thread as ready for execution. The thread will start its execution at `start_time` (second parameter) and will run at intervals specified by `period` (third parameter) given in nanoseconds. If time interval of data is 0.01 sec, then period is given by *pthread_make_periodic_np(pthread_self(),gethrtime(),10000000).*

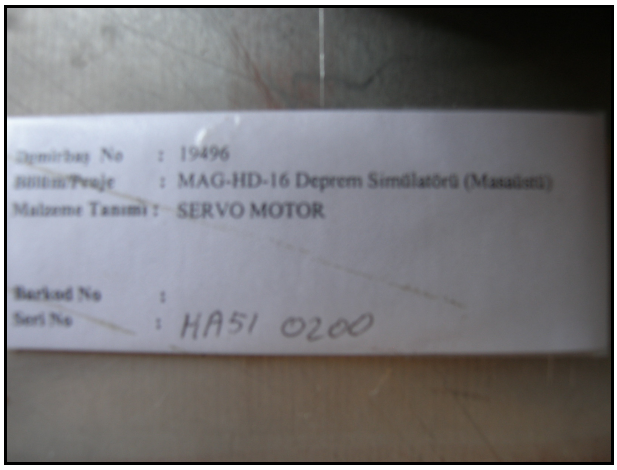


Figure A.21. Label on SHAKE1

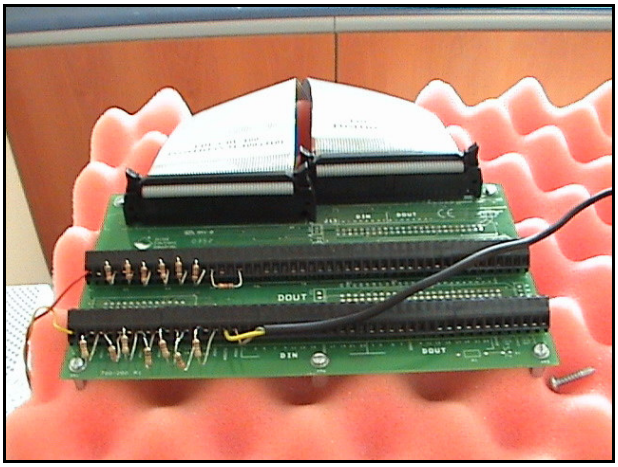


Figure A.22. Outer connection parts of PowerDAQ