DESIGN OF A TMS320 C 25 SIGNAL PROCESSOR FOR USE IN A MONOPULSE RADAR

C P van der Linden

August 1990

Submitted to the University of Cape Town in partial fulfilment for the degree of Master of Science in Engineering

I, C P van der Linden, submit this half thesis in partial fulfilment of the requirements for the degree of Master of Science in Engineering. I claim that this is my original work and that it has not been submitted in this or in a similar form for a degree at any University.

ABSTRACT

The advent of better and faster digital signal processing chips has led to digital implementation of many functions that have previously only been possible using analogue techniques. One such field is monopulse radar where available processing time is limited strictly to the radar pulse repetition frequency. The aim of this thesis is to design a specific signal processor using a Texas Instruments TMS320 C25 processor. This design is intended for monopulse radar systems using low pulse repetition frequencies.

Features typical to monopulse radar signal processing, have been described here as system requirements. From this description a system specification, which is in fact the functional design of the processor, has been developed. Prototype circuitry was then designed and built in order to test the feasibility of performing, within the required time, the functions outlined in the system specification. Following on from the results of the tests, design of the hardware commenced.

The design was successfully completed and tested. Although the TMS320 C25 was not found to be the ideal processor for this application, it is capable of performing the task within the required time. Careful consideration was given to the software design. A trade off between easily maintainable, high level language software, and high speed assembler had to be made. The final product is written in C but with critical procedures implemented in in-line assembler.

This thesis provides insight into the type of hardware and the level of signal processing required for one type of signal processor used in a low PRF monopulse system.

ACKNOWLEDGEMENTS

I would like to thank ESD South for allowing me the opportunity to conduct this thesis and for the technical support and guidance extended to me.

TABLE OF CONTENTS

List of figures

List of tables

NOMENCLATURE

AGC	Automatic gain control
CFAR	Constant false alarm rate
DSP	Digital signal processing
ECI	Estimation Channel Interface
FIFO	First In First Out
I/O	Input and/or Output
MB	Multibus
MTI	Moving target indication
PRF	Pulse repetition frequency
PRI	Pulse repetition interval
RDP	Radar Data Processor
SNR	Signal to noise ratio
SP	Signal Processor
LSB	Least significant bit

GLOSSARY

Detection counter

This shall be a hardware counter in the estimation channel interface (ECI), which shall count the number of detections that occur in the range search window. Each PRI this counter shall be reset. As each range bin is processed the CFAR (constant false alarm rate) detector shall indicate to the ECI whether a detection has occurred in that bin or not. If a detection occurs the detection counter is incremented. There is an overflow bit which is set if the counter overflows.

Range bins

Range bins shall be the term used for the range time divisions. Bins shall be equivalent to Range Gates as defined by Skolnik[1].

Range gate

Each PRI, samples are sent down the estimation channel. These samples should ideally be centred around the mid point of the return pulse and should be spaced so that the entire return pulse can be sampled. The range gate shall thus represent the range resolution of the radar. The range tracking mechanism attempts to keep the centre of the range gate positioned over the centre of the return pulse.

Range gate register

The range gate register shall be a register in the estimation channel interface (ECI), to which the current range gate is written by the signal processor (SP). The range gate register shall be used in producing the target-in-gate signal on the ECI.

Range register

The range register shall be a FIFO register in the estimation channel interface (ECI), which shall record the ranges at which detections occur. This register shall be reset each PRI and shall have an empty flag available to indicate whether it contains data or not.

Range search window

The range search window shall be an adjustable range window under the control of the radar data processor (RDP), and shall be used to produce a target-in-window signal. This signal is important during acquisition and tracking in that it indicates to the RDP that the beam is illuminating a target i.e. angle coordinates and the range estimates are correct.

Samples

Samples shall mean estimation channel samples as opposed to range bins.

Tracking gate

The tracking gate is equivalent to the range gate and thus shall be a window of range bins which determine the target-in-gate signal. i.e. If the range gate is positioned at bin number 500, the target-in-gate signal shall be valid for bins 500 - N and 500 + N, where 2N is the range resolution of the radar. This signal can be used by the RDP for monitoring the tracking. On the loss of the target-in-gate signal the RDP will decide on whether to resort back to acquisition or not.

1 Scope

1.1 Purpose

The purpose of this thesis is to design a signal processor, using a TMS320 C25 intergrated circuit, for a radar system implementing one form of monopulse.

1.2 Introduction

This thesis is a report on the design of the hardware of a signal processor. It is laid out to show the method of approach and the development of the design.

Section 2, "Design requirements", describes the basic features of a signal processor for monopulse radar systems. These requirements were derived by examining a typical monopulse system. From this a decision was made as to how the signal processor would best fit within a specific system.

Section 3, "System specification", contains the functional design of the processor. All processes to be executed by the processor are specified functionally here. It was intended that the hardware design should follow directly from this specification.

In order to minimise the risk in the final product, it was decided to build prototype circuitry for hardware and software optimisation. In order to optimise either hardware or software, one has to be very familiar with the circuitry and development environment. The test circuits were used to develop the necessary familiarity. Section 4 describes this process of optimisation.

In section 5, the hardware design is presented. Included, is a functional description which describes how, all the functions mentioned in section 4, are implemented in the hardware.

Finally, a brief report is given on tests performed on the completed signal processor. These tests were done before the processor was integrated with a radar system, but after detailed checks on the hardware and software were carried out.

2 Design requirements

2.1 General

The fundamental design of the signal processor (SP) must adhere to a specification of a generic monopulse system. In addition to this, a list of specific requirements must also be adhered to, which cater for the requirements of a particular radar in which the SP will function.

The generic monopulse specification presented here is taken from Sherman's "Baseline Monopulse Radar". It is not the main aim of this dissertation to become involved in the development of radar systems, but rather in the practical implementation of various functions within a system. Therefore only a brief outline is provided here. Readers requiring more information should consult Sherman [5].

2.2 Generic monopulse radar specification

2.2.1 Functional overview

Figure 1 is a functional diagram which shows in simplified form the major functions within a system. This diagram has been slightly modified in order to indicate the functions of the SP. A feature common to all monopulse systems is that they respond to voltage ratios or phase differences or both. This diagram is suitable for amplitude or phase comparison systems.

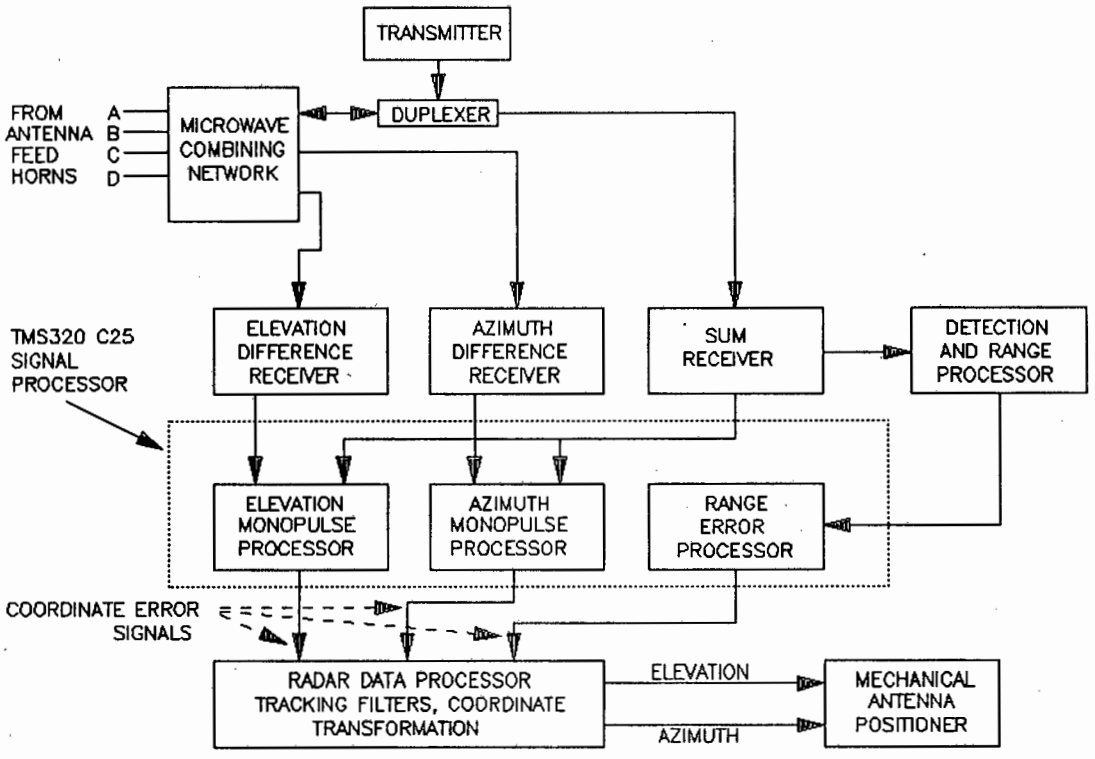


Figure 1. Functional diagram of a monopulse radar

2.2.2 Angle tracking

Angle errors in monopulse systems may obtained by using various methods such as amplitude or phase comparison techniques. In an amplitude comparison system, two overlapping beams are required in each of the azimuth and elevation planes. These beams may be generated by four adjacent horns illuminated by a single antenna. From the ratios of the amplitudes, the angle of target to boresight can be determined. A target directly on the axis of symmetry would yield equal amplitudes in all four beams. Figure 2 [4] shows two antenna beams and their gain, sum and difference patterns.

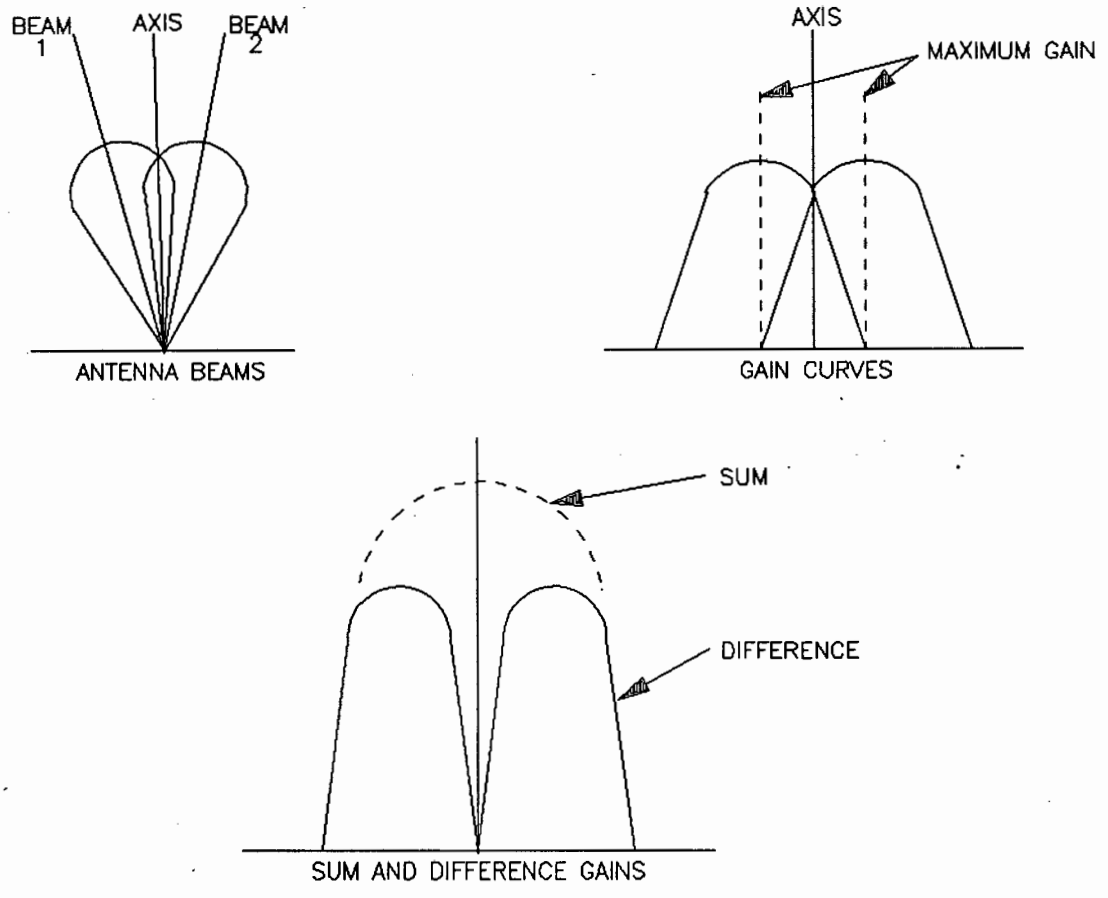


Figure 2. Monopulse beams and sum and difference patterns

All monopulse processors are intended to produce outputs that depend on ratios, and not absolute values, of signal voltages [5]. The ratio d/s (difference/sum) is the complex ratio where

$$d=|d|\exp[j\delta_d] \qquad s=|s|\exp[j\delta_d] \tag{1}$$

The ratio expressed in I and Q format is [5]

$$\frac{d}{s}=\frac{d_i s_i+d_q s_q+j(d_q s_i-d_i s_q)}{s_i^2+s_q^2} \tag{2}$$

where

$$\begin{aligned} d_i &= |d| COS(\delta_d) = Re(d) \\ d_q &= |d| SIN(\delta_d) = Im(d) \\ s_i &= |s| COS(\delta_s) = Re(s) \\ s_q &= |s| SIN(\delta_s) = Im(s) \end{aligned} \qquad (3)$$

Thus

$$\begin{aligned} Re\left(\frac{d}{s}\right) &= \frac{d_i s_i + d_q s_q}{s_i^2 + s_q^2} \\ Im\left(\frac{d}{s}\right) &= \frac{d_q s_i - d_i s_q}{s_i^2 + s_q^2} \end{aligned} \qquad (4)$$

Figure 3 shows the sum and difference patterns and the ratio d/s, as a function of boresight to target angle [4].

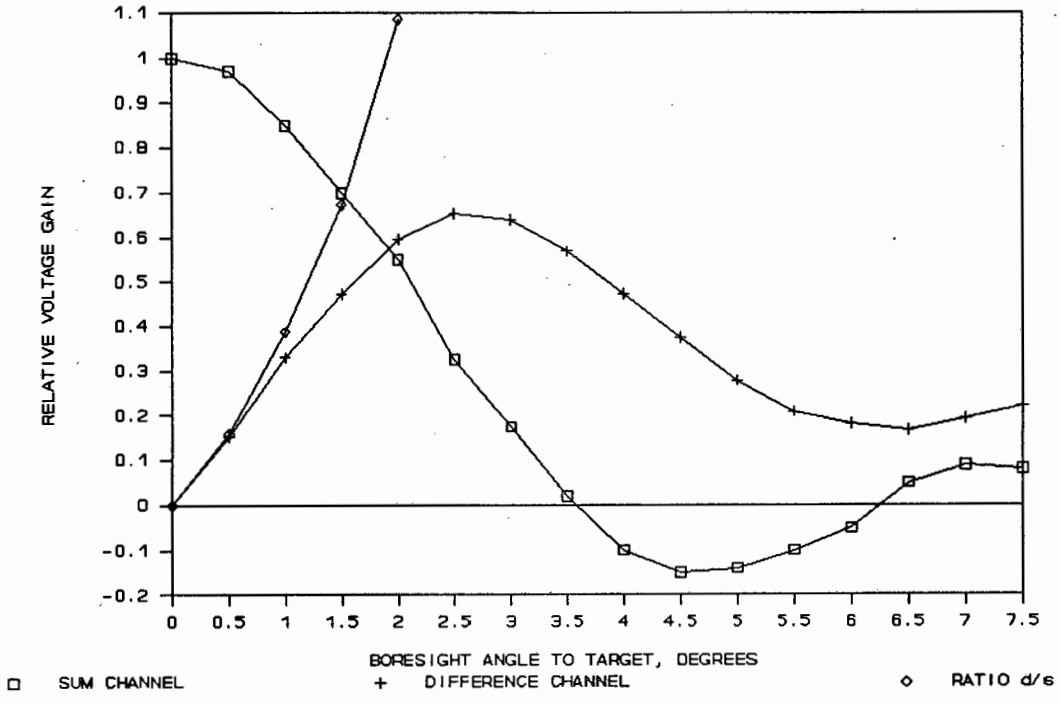


Figure 3. Sum, difference and ratio d/s patterns

The output in the two angle coordinates, consisting of the voltage amplitude ratio shown above, can be

termed the normalised difference signals. These difference signals may now be fed into a tracking filter, usually an adaptive alpha beta filter [1]. The output of the tracking filter is further processed, eg. coordinate transform, before being output to the antenna positioner.

2.2.3 Range tracking

The sum channel is used for range tracking. One possible method of producing a range error is the use of the split gate calculation. In this calculation a range discriminant x is calculated as follows:

$$x = (\text{early} - \text{late})/(\text{early} + \text{late})$$

where early and late refer to early and late samples as shown in figure 4.

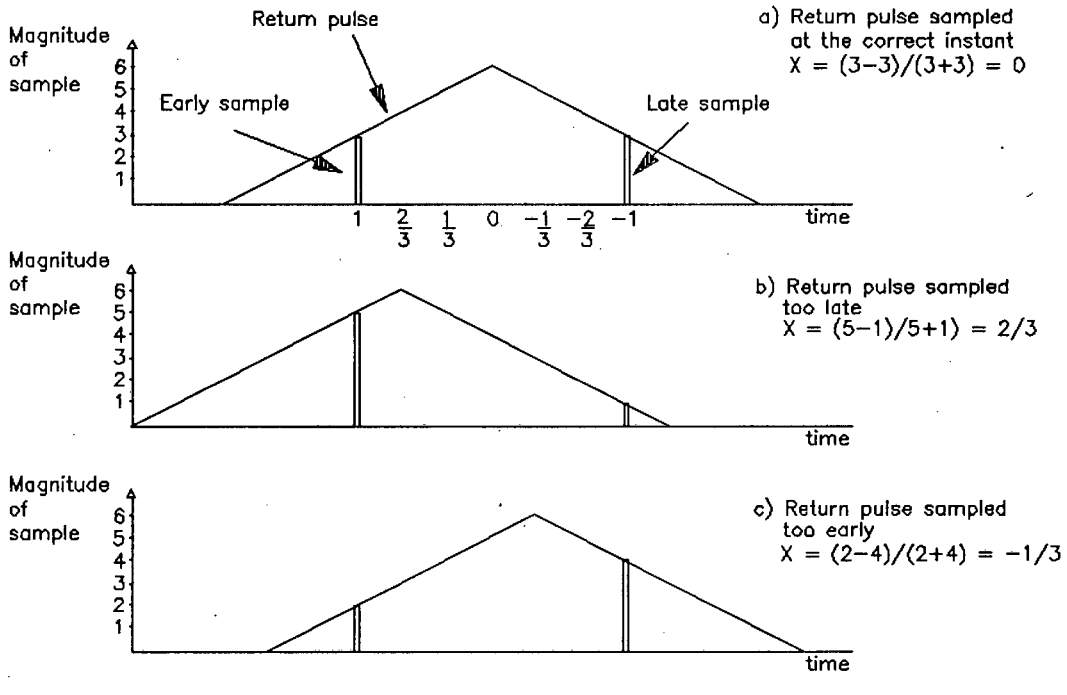


Figure 4. Return pulse and split gate calculation.

The aim of the tracking system will be to keep the target exactly on range, indicated by a discriminant

of 0. The discriminant varies between +1 and -1 indicating where the target lies in time with respect to the sum channel. The discriminant is thus an error signal which can be fed into a tracking filter, usually an adaptive alpha beta filter. The output of the filter can be used to position the range gate.

2.3 Specific design requirements

A list of design requirements is needed as a base, from which the signal processor can be developed. A particular system has been examined and its requirements for an estimation signal processor have been determined. A list of these requirements is included in appendix A and is summarised in this chapter.

The details of the system on which the specific requirements are based, are company confidential. It is thus intended that the system information described below should form sufficient background, without direct reference to any other specification, for the design of the SP.

a. General description

The SP must cater for low PRFs of up to 3 kHz. Software functions that occur in this particular radar may be divided into two groups: functions that need to be executed each PRI, and functions which are not bound to the PRI at all. As a general guideline the SP must be able to perform all the software functions that occur within each PRI. The SP must perform post detection integration. It must therefore integrate data over some slower-than-PRF rate. The integrated data must be made available at the end of each integration period. This period will be 20 ms. The SP will thus have two timing inputs, PRI and 20 ms synchronisation pulses.

Each PRI the SP must execute a sequence of functions in a specific order. On receiving raw radar data it must multiply the samples with calibration factors. These factors, which compensate for receiver channel imbalances, will be generated after power-up and when the radar is not operational,. The SP must then implement an MTI filter in the form of a triple pulse canceller [1]. The output of the MTI filter will then be used for monopulse and split gate calculations. The number of PRIs occurring in 20 ms depends on the PRF. The results of the monopulse and split gate calculations must be averaged over all the PRIs for each 20 ms period.

Furthermore an AGC function must be implemented. This function will use the STC facility to select different receiver gain settings.

b. Interface to the SP

Communication with the SP will occur via three mechanisms. Firstly, the SP will have a direct link to an interface card, called the estimation channel interface (ECI). The ECI card interfaces the estimation channel (a serial channel) to the SP. The SP will receive all the radar data via this link. Secondly, the SP will have a series of ports accessible via the Multibus II (or MB) iPSB interconnect and I/O data spaces [6]. The SP will receive all instructions and send all data resulting from calculations via the MB. Lastly, the SP will have an output port dedicated to the output of radar evaluation data.

c. Control of the SP

The underlying philosophy of the SP software will be that the SP is a slave to the controlling processor, the

Radar data processor (RDP). The SP must inform the RDP of its current status. If an error occurs the SP must merely report this to the RDP. Appropriate action will then be taken by the RDP. The RDP will thus issue a SP command word and will read a SP status word every 20 ms.

3 Signal processor system specification

In this section the system specification of the signal processor is described. This system specification must fulfil the design requirements specified in section 3.

3.1 Context

The SP shall be part of a monopulse radar system. Figure 5 shows a physical context diagram of the SP.

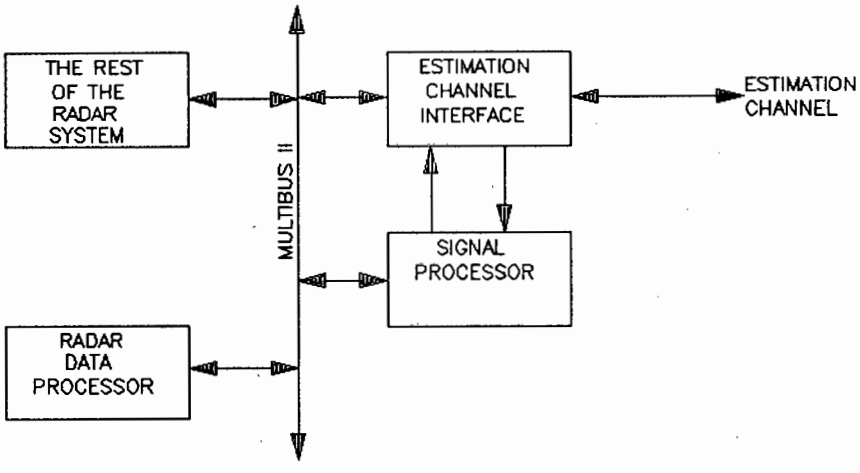


Figure 5. Context block diagram

3.2 Purpose of the SP

The radar tracking function shall be implemented in two units, the SP and the ECI. The ECI shall perform certain hardware functions. The purpose of the SP is thus to provide software intelligence to the high speed (PRF) tracking process.

3.3 Basic Functions of the SP

Functions within the SP shall be mode dependent. Modes can be categorised as acquisition, tracking and setup. Listed below according to mode are the basic functions of the SP. These are divided up into PRI based functions and 20 ms based functions.

3.3.1 Acquisition mode

Figure 6 shows a functional block diagram of the SP during acquisition mode.

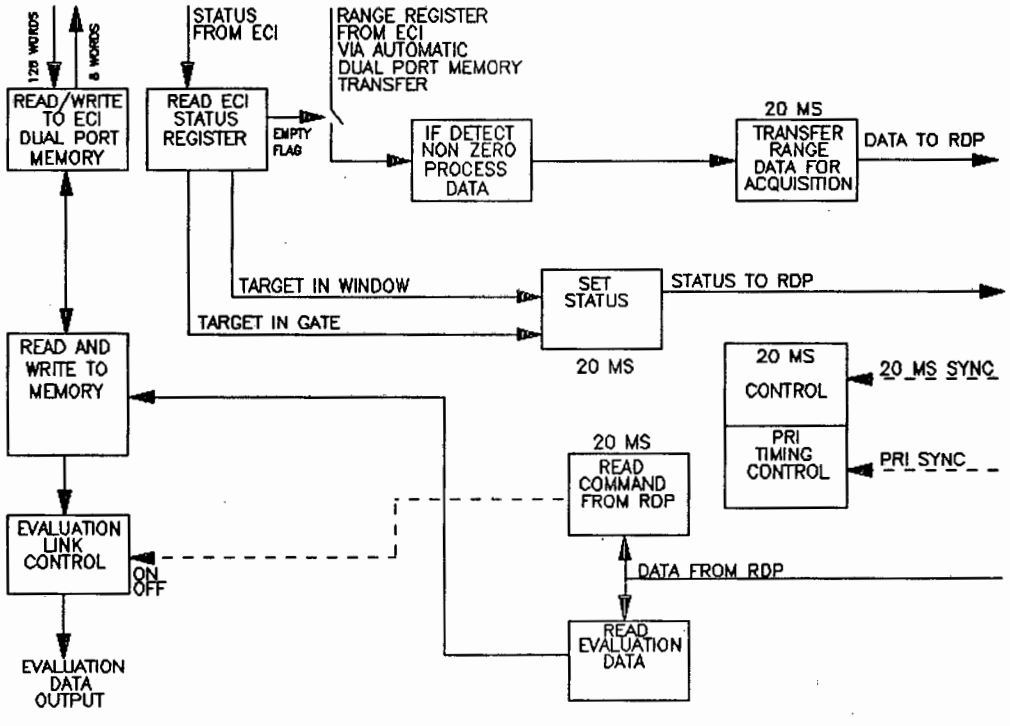


Figure 6. Functional block diagram during acquisition

3.3.1.1 PRI based functions

a. Monitor detection status

Each PRI, the SP shall check the detection counter. While there is no target indicated, the counter shall

be zero and the SP shall continue to monitor this without doing anything else.

b. Read range register

When the range register empty flag no longer indicates empty the SP shall read the range register. The range register shall contain a maximum of 64 ranges.

c. Fading or multiple targets

Targets that are detected after a previous target has faded shall be treated as the current target. i.e. If a target fades but is not replaced by another target, its range shall be retained in memory. If a second target at the same or a different range is detected the details of the first target shall be replaced by the new one.

3.3.1.2 Functions based on 20 ms

a. System timing

The system timing shall be controlled by the system 20 ms timing pulse. This shall be implemented using hardware interrupts.

b. Write data to RDP

On receiving the 20 ms interrupt the SP shall calculate the average range at which the target was detected. This shall be an arithmetic mean i.e. if the target was present for 20 PRIs then the ranges for these 20 PRIs shall be added up and divided by 20. The result shall be output to the RDP via the multibus.

FIFOs shall be used to eliminate the need for handshaking between the RDP and the SP. The SP shall also update the status register which contains the target-in-gate and target-in-window signals.

c. RDP read timing

After the 20 ms signal the RDP shall wait a minimum time of 660 micro seconds before reading from the SP. This is to allow the SP time to prepare the data.

3.3.2 Tracking mode

Figure 7 shows a functional block diagram of the SP during tracking mode.

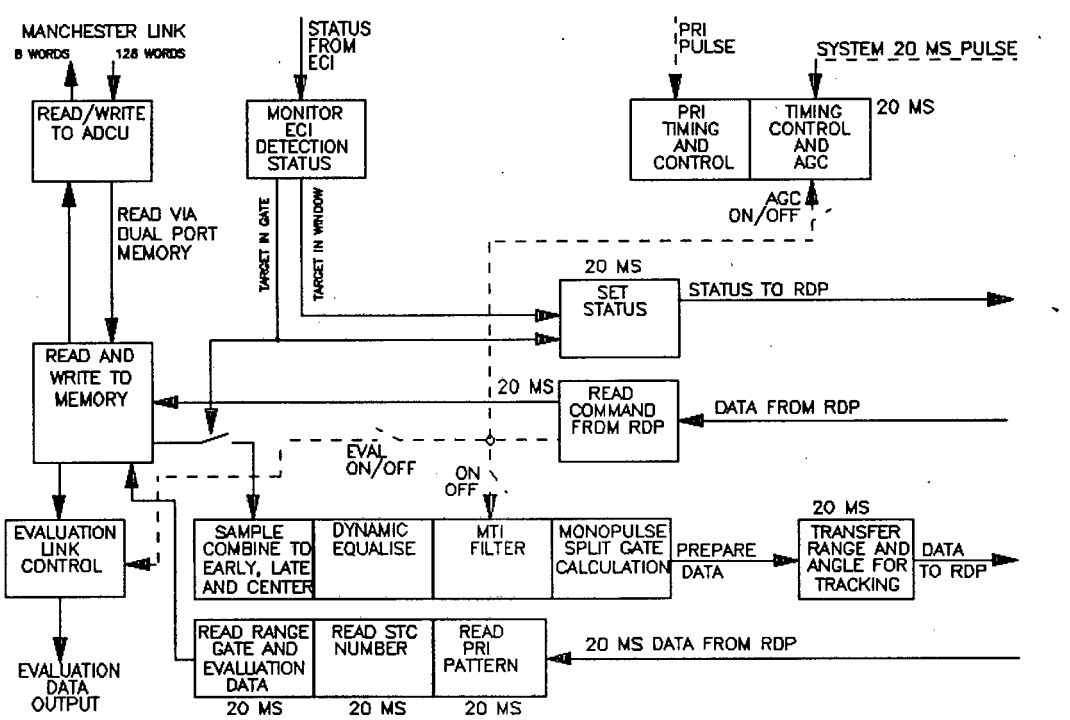


Figure 7. Functional block diagram during tracking

3.3.2.1 PRI based functions

a. Read sample data from ECI

Each PRI, the ADC sends radar data to the ECI. When all the data has been received by the ECI, the ECI shall send a data ready pulse to the SP. On receiving this pulse the SP shall read all the data from the ECI.

b. Write ADC command data to ECI

After reading from the ECI the SP shall then write eight ADC command words to the ECI. After writing these words to the ECI the SP shall acknowledge this by setting an acknowledge bit which shall indicate to the ECI to transmit the data to the ADC.

c. Check the test word

Each PRI the SP shall send a test word to the ADC. The ADC shall send this word back to the SP. Corruption of this word will indicate a fault in the estimation channel. The SP shall thus check the returned test word each PRI and if a fault occurs then an error is indicated to the RDP.

d. Read the ECI status register

The SP shall read the ECI status register in order to monitor the target-in-gate signal. If there is no target-in-gate processing of the samples shall not take place. If the target-in-gate signal is present, then the processing of data shall proceed as outlined below.

e. Process data

The sample data is now processed. The processing steps are: sample combine, receiver equalisation, MTI, monopulse calculation, range split gate calculation, AGC and PRI select. The split gate and monopulse results are stored in memory.

Sample combine. The received data is reduced to three samples; these being an early sample, a centre sample and a late sample.

Equalisation. Dynamic equalisation is performed on the incoming samples in order to compensate for channel imbalances. The equalisation is achieved by multiplication of the samples by previously calculated calibration factors which are stored in memory.

MTI. If the MTI is ON then the software shall implement a triple pulse canceller MTI.

Monopulse calculation. This calculation is the division of the azimuth and elevation channels by the sum channel. The result of this division shall be a value proportional to the angle off boresight. The proportionality constant is given by the slope of the detection curve and is dependent on receiver parameters. i.e. (K * Ve/Vs) = elevation angular error in radians and (K * Va/Vs) = azimuth angular error in radians.

Split gate calculation. This calculation takes the form X = (EARLY-LATE)/(EARLY +LATE). The value of X shall thus represent the distance from the middle of the tracking gate to the middle of the theoretical

return pulse (see section 2 for an explanation of the split gate calculation).

AGC. If the AGC function is selected ON then the energy in the entire return pulse is calculated by adding all the samples together. If the result is lower than a specified limit or higher than a specified limit then the next STC curve with greater or lesser gain respectively is selected. The energy measurement shall be input to a first order filter with a variable response time. The selection of the STC curves is based on the output of this filter. If the AGC is switched OFF then the required STC curve shall be selected by the RDP each 20 ms.

PRI select. The SP shall send a control word to the ADC each PRI to control the PRI. eg. If half PRF is required during testing then the appropriate word can be sent to achieve this.

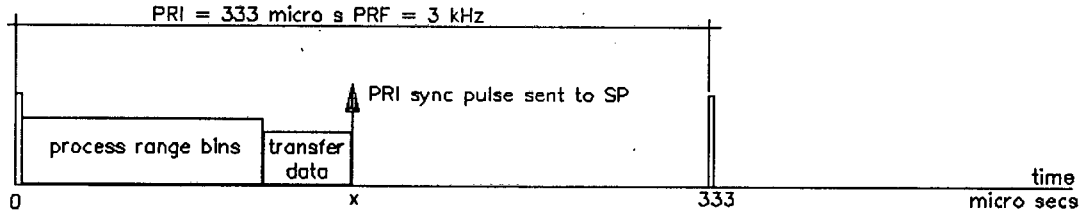
f. Idle time

After the processing of the data is complete the SP shall idle. Figure 8 on page 18, shows SP process timing. The idle is terminated by the PRI or 20 ms timing signals.

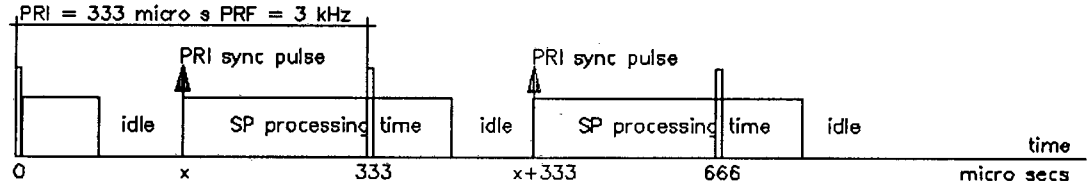
3.3.2.2 Functions based on 20 ms

a. Timing

Timing shall be controlled by the system 20 ms pulse which shall be connected to an interrupt on the SP. On receiving this interrupt the SP shall commence preparation of data for the RDP.



(a) Hardware processing resulting in the PRI sync pulse which is fed to the SP. Note that the data transfer refers to the data transfer to the ECI



(b) SP processing timing

Figure 8. SP process timing.

b. Maintenance of MTI filters

During preparation of RDP data the normal processing of estimation data will cease. The MTI filters, however, must be continuous and therefore the PRI interrupt must still be active. On receiving a PRI interrupt the SP will read in estimation data as normal but shall only equalise it and add it to the MTI history in memory. After this it will return to the preparation of data for the RDP.

c. RDP read timing

The preparation of data shall take no longer than 2 PRIs i.e. the monopulse and split gate calculations will cease for no longer than 2 PRIs. The RDP will thus wait for a minimum time of approximately 660 micro seconds, before reading the data from the SP.

To eliminate the need for handshaking the SP will make use of FIFOs for the data transfer to and from the RDP.

d. Preparation of data for the RDP

There shall be a variation of PRIs per 20 ms up to a maximum of 60 PRIs (PRF = 3 KHz), depending on the PRI stagger pattern and PRF. Each PRI, the SP calculates part of the monopulse and split gate calculations and stores these in memory. On receiving the 20 ms timing pulse the remainder of the calculations are performed on the arithmetic mean of the values stored in memory.

e. Transfer of other target details

Other data that must be sent to the RDP are the contents of the range register i.e. positions of detections within the range search window. The SP will process the contents of the range register The positions of valid targets are then made available to the RDP. The positions of a target will be indicated by a range bin start and stop number.

f. Read SP status register

The status register will also be updated by the SP in time to be read by the RDP simultaneously with the radar data. In this status word are the target-in-gate and target-in-window signals. These signals will represent the status of the target-in-window/gate signals on the ECI in the last PRI before the 20 ms interrupt.

g. Evaluation data from the RDP to the SP

The current estimated position, the next predicted position, the calculated target rates, the current positioner coordinates and the positioner velocities form part of the information to be output to the evaluation port. The evaluation port will be located on the SP and thus this data must be sent from the RDP to the SP each 20 ms. On receiving the 20 ms system synchronisation pulse, the RDP must send this data to the SP. There is no minimum time delay between the 20 ms pulse and this process.

3.3.3 General requirements

a. Evaluation link

The SP shall be responsible for the evaluation link which will be a high speed parallel link. The purpose of the link will be to evaluate the radar and to provide debugging facilities. Radar data will be output via this link to a recording device, probably an IBM PC or compatible, and this data will be analysed in conjunction with a recording of the video picture. Only 20 ms information will be of interest when using the video picture but the PRI data within that 20 ms period will also be available.

When evaluation information is required, the SP will send radar data to this port at a rate of 10 words per PRI. The evaluation link will be synchronised with a recording of the radar video picture. This will be achieved by including in the data output the 20 ms period number and the PRI number within that 20 ms period. The 20 ms periods will be numbered from zero when the evaluation link is selected ON. The video picture shall also have 20 ms timing information thus

the radar data for a particular video picture will thus be obtained by looking for the data linked to the same 20 ms number.

b. Power up

The SP shall on power-up, or in response to a request from the RDP (in self test or maintenance mode), perform several internal system checks eg. the RAM and the ROM and any other checks that may aid in establishing the functionality of the SP. The status of the SP must then be made available to the RDP via the MB.

c. Testing

When requested by the RDP, the SP will accept test data from the multibus instead of data from the ECI. This data will be used for checking hardware and software functions in the SP. Results of this checking will then be output via the Multibus to the RDP for evaluation.

The SP will also continuously monitor the test word sent to the ADC and back. Corruption of this test word will indicate a fault in the estimation channel.

3.4 Interface definition

In order for the SP to be compatible with other equipment, both hardware and software require to be carefully interfaced. Hardware interfacing is achieved by using I/O ports with latches and FIFOs etc. Software interfacing requires the definition of a detailed data structure. Data is then passed between different types of equipment within the framework of the data structure. The data structure is thus the definition of each word for each I/O port in all modes.

In the SP some ports have a fixed meaning eg. the status port only outputs the status word. These ports require bit definition i.e. each bit has a particular meaning. The SP data structure can be seen in appendix B.

3.5 SP Characteristics

3.5.1 Performance characteristics.

a. PRF range

The SP shall be able to perform the software routines during tracking mode for PRIs of up to 3 kHz. During tracking mode the software takes up the most time and thus is critical for time analysis.

b. Processor

A TMS 320 C25 digital signal processor running at 40 MHz (exact frequency not critical) shall be used if it can fulfil the time requirements. The cycle time of the TMS320 C25 is 100 ns.

c. Memory

ROM : A minimum of 12 kbyte by 16 bit wide on-board ROM for program memory shall be provided for. The ROM shall be zero wait state i.e. less than 40 ns access time.

RAM : A minimum of 16 kbyte by 16 bit wide on-board RAM shall be installed. The RAM shall be zero wait state i.e. less than 40 ns access time.

d. Logic

All inputs and outputs connected to the SP shall be TTL compatible.

e. Sample resolution

The estimation channel samples will have a resolution of 10 bits. The SP shall read these 10 bits in as a 16 bit, sign extended, 2's compliment number. The sign extension will be done in the hardware. Provision will be made for the resolution to change from 10 to 12 bits by ensuring that the tracks on the PCB for the sign extension of the upper two bits can be cut if required.

f. Output resolution

All radar data output will be 16 bit 2's complement words except where higher precision is required. In this case a floating point format using two 16 bit words will be used. Status and command words will be either 8 or 16 bit words as defined in this document.

g. Initialisation

On power-up the SP will reset itself completely. Provision shall also be made for the RDP to invoke a reset by selecting the reset code in the status word. The reset command will be encoded in the test mode control bits.

h. Watch dog

There shall be provision for a watch dog timer on the SP. This timer will cause an SP reset if it is not toggled within 150 ms.

i. Manual reset

There shall be a manual reset button on the front panel of the SP.

j. Running indication

An LED mounted on the front panel of the SP will indicate that the processor is running. This LED will be coupled up to the toggling bit on the watch dog. The LED will thus indicate when the processor has stopped toggling the watch dog.

3.5.2 Physical characteristics

a. PC Board

The SP will consist of one standard double Eurocard PCB which shall plug into the multibus II chassis and motherboard.

b. Connectors

The SP PCB will have two 96 pin DIN 41612 type connectors. One will be used for the multibus II iPSB parallel bus motherboard and the other for the custom made back plane board for communication to the ECI.

4 Test hardware and software and optimisation

It is only when the initial design of a system is complete that optimisation of that system can begin. Both hardware and software was examined for possible optimisation.

4.1 Aim of the test circuitry

In order to minimise the risk of the final product, hardware for testing was built. The objectives of the test circuitry were as follows:

a. To become familiar with the hardware requirements of the TMS320C25.
b. To become familiar with the TMS320 development environment. This consists mainly of the Texas Instruments XDS/22 in-circuit-emulator controlled via a serial link from an IBM compatible PC.
c. To become familiar with the TMS320 assembler, the TMS 320C25 software simulator, and a C compiler. These tools will be used to test software.
d. To perform vital speed tests. The test circuit must represent the final circuit closely enough for the speed tests to be meaningful.
e. To evaluate the TMS 320 C compiler.

4.2 Software optimisation

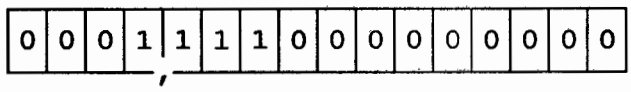
The following points describe selected parts of the process software that have been modified. Modifications have been made to enhance either the speed of execution or the quality of the radar information.

4.2.1 Number representation

The initial intention was to represent all large numbers in floating point format. Large numbers in this case refers to numbers that are larger than 16 bits. A major disadvantage of the TMS320 C25 is that it does not support floating point arithmetic. To overcome this a floating point subroutine was implemented. Guidelines to floating point arithmetic are shown in the Texas Instruments Software Application notes [7].

Once a decision to use floating point has been made, all further arithmetic with that data must be done using special routines. If any data is to be output, then it must be ensured that the processor receiving the data is compatible with the floating point format.

It was soon discovered that the floating point calculations take excessive time to execute. An alternative data format was then implemented. Numbers that are too large to fit into a sixteen bit word are represented by two sixteen bit integers. It is a much easier and quicker task to manipulate sixteen bit integers than to do floating point calculations. Numbers for which high precision is necessary and whose values are in the region of one, are represented in a 12 bit format. In 12 bit format the first twelve LSB bits are taken as after the decimal comma. For example, the number 1.875 is represented in 12 bit format as shown below.



Numbers in the region of one are mostly required in equalisation. Samples are received from the ADC and are calibrated in the SP to compensate for receiver channel imbalances. These calibration factors have values very close to unity.

4.2.2 Avoiding unnecessary division

The TMS320 C25 does not provide a built in divide command in its assembly language. Both the monopulse and the split gate calculations involve division. To cater for this, a division subroutine was implemented. As with floating point calculations, divisions take a long time to execute.

To circumvent the division, the angle and range error signals are now each represented as a two word numerator and a two word denominator. These values are output in this format to the RDP, a processor which supports full floating point arithmetic. The first thing the RDP must do is the division. Thus, without loss of information or accuracy, the divisions can be entirely avoided in the SP.

4.2.3 Method of radar signal integration

Each PRI radar data is sent down the estimation channel. The error signals resulting from each PRI are only used every 20 ms. Thus, the PRI data is integrated during each 20 ms period i.e. post-detection integration[1].

The monopulse calculation is of the form Vd/Vs. Vd and Vs represent the amplitude of the delta and sum channels respectively. It was initially intended to perform the division each PRI. The results of these divisions were then to be added each PRI, and an

arithmetic mean calculated every 20 ms. There are two disadvantages with this method:

a. The aim of integration is to reduce the effect of random noise. By adding noisy quantities together, the thermal noise component will tend to zero while the signal component will grow. Division is a non-linear process. Dividing noisy signals can result in spurious spikes of considerable amplitude. Although these spikes would be absorbed in the averaging process, they adversely (because of the division) affect the final result.
b. The division has to be executed every PRI. This is a time consuming process.

An alternative method is recommended by Sherman[6]. In this method a weighted average is calculated. Each measurement is weighted by the ratio of the sum signal power during that measurement, to the total of the sum signal powers during all measurements. In this way, a greater weight is allocated to measurements that occur when the signal strength is strong. The formula for calculating the weighted average is as follows.

$$Re(\phi) = \frac{\sum_{k=1}^{n} |d_k||s_k|\cos\delta_k}{\sum_{k=1}^{n} |s_k|^2} \qquad (5)$$

where Re(ϕ) = real part of the indicated angle
d = amplitude of the delta channel
s = amplitude of the sum channel
n = number of PRIs
δ = relative phase between channels

There are three advantages in using this method. Firstly, the averaging is done before the division. Secondly, the division need only be done each 20 ms. This enables the division to be entirely avoided,as described in section 5.2.2. Finally, by giving more weight to samples that are received when the signal strength is stronger, a better result is obtained.

4.2.4 Manual optimisation of compiled code

A question that has persisted throughout the design of the SP software is, how efficient is the C compiler? Initial software tests using the compiler showed up some bugs in the I/O library routines. This cast a poor light on the performance of the compiler. Once identified, these problems were easily overcome by using simple custom written I/O routines.

Subsequent examination of the code produced by the compiler has revealed the code to be acceptably efficient. There is, however, plenty of scope for manual optimisation. One time consuming feature is the generation of a comprehensive run-time stack. This stack requires to be saved each time a subroutine is called, and this takes time. The benefit, though, is a far better program structure which is easier to maintain.

To increase speed of execution, looping in the program has been avoided if possible. Where repetitive functions were required, the function was written out as many times as is required. This made for a weaker programming structure and lead to more bulky code, but the benefit to speed was of prime importance. A simple example of this is shown in the C routine below. The routine sets up a loop which

inputs a value from port zero and immediately outputs the same value to port one. The loop is executed three times.

```
/*********************************************************
* C program to set up a loop which reads a value from port
* 0 and outputs the same value to port 1. The loop is run
* 3 times.
*********************************************************/

int i;                          /* Declare an integer i */
int val;                        /* Declare an integer val */

main()                          /* Start of main routine */

{

for (i= 1; i <= 3; ++i)         /* Set up loop from 1 to 3 */
      {
      asm("  IN   _val, 0");   /* In-line code to input  */
      asm("  OUT  _val, 1");   /*val and output to port 1 */
      }

}
/*********************************************************/
```

Compiled for the TMS320 C25, the assembler for this routine is shown below. Some redundancy in the assembler can be seen in the L3 loop where a "save i" is followed directly by a "load i". The "load i" is redundant. Also in loop L3, the memory page pointer is set every time the loop is executed. This could be done once, before the loop thus saving an extra cycle during each loop. The saving becomes more significant if the loop is executed many times.

```
*******************************************************
* FUNCTION DEF : _main (TMS320 C25 assembler program)
*******************************************************
_main:
   POPD  *+              *At start of every subroutine
   SAR   AR0,*+          *the stack is initialised in this
   SAR   AR1,*           *way. The AR0 and AR1 registers
   LARK  AR0,1           *are saved and then initialised to
   LAR   AR0,*0+         *default values

   LACK  1               *Initialise the variable "i".
   LDPK  _i              *
   SACL  _i              *
   SSXM                  *Set the sign extension mode.
   LAC   _i              *Do the first test for i <= 3.
   SUBK  3               *If i >= 3 jump past the main
   BGZ   L2              *loop by branching to label L2.

L3:                      *Start of the main loop.
   IN    _val, 0         *Input value from port 0.
   OUT   _val, 1         *Output value to port 1.
   LDPK  _i              *Set memory page pointer.
   LAC   _i              *Load i.
   ADDK  1               *Increment i.
   SACL  _i              *Save i.
   LAC   _i              *Load i.
   SUBK  3               *Subtract 3 from i.
   BLEZ  L3              *Branch to L3 if result <= 0.

L2:                      *Ending the sub routine.
   MAR   * ,AR1          *Reset the stack
   SBRK  2               *
   LAR   AR0,*-          *
   PSHD  *               *
   RET                   *Return
   .end
```

To illustrate the overhead the maintenance of the stack requires, the assembler program can be replaced by the assembler shown below. This section of assembler would no longer be a sub-routine, but would be re-written when ever the input/output loop is required.

```
*********************************************************
* FUNCTION DEF : _main (TMS320 C25 assembler program)
*********************************************************
_main:
    .
    .
    LDPK _dummy          *Set the memory page pointer
    IN   _val, 0         *Input value from port 0.
    OUT  _val, 1         *Output value to port 1.
    IN   _val, 0         *Input value from port 0.
    OUT  _val, 1         *Output value to port 1.
    IN   _val, 0         *Input value from port 0.
    OUT  _val, 1         *Output value to port 1.
    .
    .
    .end
*********************************************************
```

4.2.5 Method of synchronisation

The SP is required to produce results from the monopulse and split gate calculations every 20 ms. Two methods of synchronising to the 20 ms pulse were considered.

The initial method of preparing data for each 20 ms was to record the time since the last pulse. The new data was then to be prepared just before the next 20 ms pulse. Two methods of recording time were possible. These were to either count the PRI pulses or to implement the timer function on the TMS320 C25. Counting PRI pulses required less software overhead than the timer function, and was therefore the preferred method. The procedure, then, would be to integrate data during each PRI until the PRI before the expected 20 ms pulse. In this last PRI, normal tracking operation is suspended and data is prepared for the 20 ms read from the RDP. There are two problems with this method. Firstly, the accuracy of the 20 ms period is unknown. Thus, if synchronisation between the PRI count and the 20 ms pulse is lost, there is no way in which it can be

recovered. Secondly, there is a software overhead in keeping track of the time by counting PRI pulses.

An alternative method of synchronising to the 20 ms pulse is to use the pulse as an interrupt. On receipt of this interrupt, the SP would halt normal PRI processing and then prepare the 20 ms data. There are three differences between this and the first method. Firstly, There is no software overhead in keeping synchronisation with the 20 ms pulse. Synchronisation also cannot be lost. Obviously, these are advantages. Secondly, a disadvantage is that up to two PRI may be required to prepare the data. This depends on where, in the last PRI, the 20 ms interrupt occurs. If the interrupt occurs at the start of a PRI, then that PRI must be completed before the interrupt is served. Finally, a further disadvantage is the more complicated timing requirement imposed on the RDP. This is that the RDP must wait a minimum of two PRIs after the 20 ms pulse before it can read data from the SP.

The alternative method is preferred. Both disadvantages mentioned do not compromise accuracy or practicality. The advantages are less software overhead for the SP and a more secure method of synchronisation.

4.3 Hardware optimisation

4.3.1 Test circuit description

The test circuit consists of two parts, a TMS320C25 processor board and an IBM PC interface with FIFO buffers. Circuit diagrams of the test circuits can be found in the sleeve on the back cover of this report. The processor board consists of a TMS320C25

with 40 MHz clock, 2 kbyte ROM, 16 kbyte RAM and an input and an output port. The interrupt pins are made available for 20 ms and PRI interrupt simulation.

The SP test board is shown in sheet 1. The circuit provides 2 kbytes of zero-wait-state ROM and 16 Kbyte of one-wait-state RAM. If access to the RAM forms part of a critical time test the one-wait-state must be taken into account.

The LED on the XF pin (a general purpose output pin) can be used as an indicator for testing. If the flag is set and reset at the start and end of routines which are executed a million times, the LED will indicate time in microseconds i.e. one second is equal to 1 microsecond.

The port decoding is done so as to allow any port access to activate the port enable signal. This is useful when testing actual software which has been written using the multi-port structure, which the final circuit will have i.e. if the data being read into the test circuit port is packed in the desired way, the circuit will react as though there are multiple ports, without having to modify the software.

An interface to an IBM PC or compatible, is shown on sheet 2. Data coming from the test circuit will be written into FIFOs C and D which are IDT 7202 1 Kbyte by 9 bit FIFOs. Data to be sent up to the test circuit will go via FIFOs A and B which are also IDT 7202 FIFOs.

4.3.2 Timing measurements

The type of software used in this system is essentially sequential software and what calculations there are, are easily verified by applying test data and checking the result. The critical parameter of the software is the execution time. Table 1 below shows a list of all the process blocks which form part of the critical time path. Time measurements have been taken on each module as shown.

FUNCTION	TIME micro s
a. Sample combine	24
b. Dynamic equalisation	19
c. MTI filtering	22
d. Update running PRI based history	18
e. Split gate calculation	50
f. Monopulse calculation	13
g. PRI select	14
h. AGC, response filter update	7
i. Evaluation data output	15
j. Data in and out to estimation channel	106
k. Data in and out to RDP	30
TOTAL	318

Table 1. Prototype process execution time measurements

4.3.3 Timing analysis

As can be seen from table 1 the required processing does fit into the required 333 micro seconds (PRF = 3 kHz), but there is not much room for flexibility. It is likely that the code, which has been written in C with some in-line assembler, could be further optimised, but there is no guarantee that this will yield substantial reduction in execution times.

From the table it is clear that nearly a third of the available time is used on reading data from the ECI.

If this procedure could be done in parallel with the SP, there would be a sufficient factor of safety generated in processing time.

4.3.4 Dual port memory

With the use of a dual port memory and the appropriate addressing hardware, the data from the ECI could be read into SP memory concurrently with normal SP processing. The hardware could be arranged so that the PRI interrupt received from the ECI would no longer trigger the PRI interrupt on the SP, but instead would start the automatic loading of the data into memory. When the loading of the data is complete, a new interrupt would be generated for the SP in order to commence PRI processing.

128 words of data, are read in from the estimation channel and 64 words from the range register FIFO. The hardware loading should occur at 10 MHz and will thus not upset the overall PRI timing by more than 20 micro seconds (192 words transferred at 100 ns per word will take 19.2 micro seconds).

4.3.5 Test conclusion

From the test it is clear that there is not enough spare processing time. It will be necessary to implement dual port memory to achieve loading of ECI data concurrently with SP processing.

5 Presentation of final circuit

5.1 General overview of the SP

The SP will be a PCB with a processor, ROM, RAM and a communication structure in the form of input and output ports. Figure 9 shows a physical block diagram of the SP. Circuit diagrams (sheets 1 to 4) are included in the sleeve on the back cover of this report.

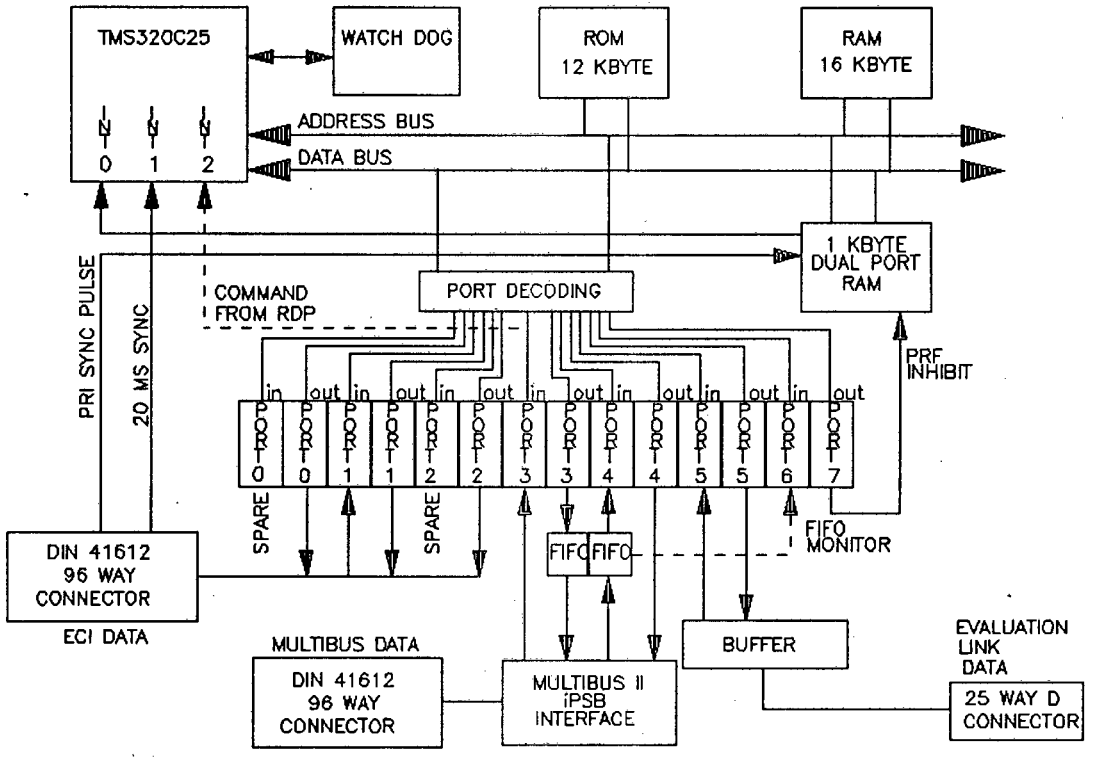


Figure 9. Physical block diagram of the SP

5.2 Memory requirements

The SP will provide the facility for 12 kbyte UV erasable PROM. This EPROM will be zero-wait-state and

this implies memory with less than 40 ns access time. The SP will also have 16 kbyte zero wait state RAM. This size RAM was chosen because it is easily implementable, using four 16 kbyte by four bit RAM integrated circuits. 16 kbyte is more than enough to store all the calibration tables, variables and stored data. The TMS 320C25 has some on-chip RAM which can also be used if it is required.

5.3 Timing

The SP will be totally interrupt driven. There will be two interrupts, one from the system 20 ms clock and the other from the ECI. This latter indicating the next PRI, which will have the highest priority. The TMS 320C25 will run on a 40 MHz clock which is generated by a TXO400 TTL clock oscillator on the SP board.

5.4 Self test functions

The SP will provide certain types of self diagnostic information. Some of this information is available on the SP status word all the time and some only on-power up or by request from the RDP. The following will be available:

a. TMS clock active. This will be the output from a watch-dog timer and will indicate that the TMS processor is running. This signal will be present on the status word all the time.
b. PRI synchronisation active. This will indicate that the SP is receiving the PRI interrupts. This signal will be available on the status word all the time.
c. Software self test. This will involve RAM and ROM tests, the results of which will be reported on the error code within the status word.

d. Tracking software test. Instead of the SP receiving data from the ECI data it will receive data from the RDP via the multibus. This data can be then be processed by some of the normal tracking routines i.e. MTI filtering, monopulse calculation and split gate calculation. Because the MTI is a triple pulse canceller, at least three sets of data should be available. These tests will not check any of the ECI communications or functions. Thus real tracking and acquisition can not be tested without the rest of the radar operating.

e. ECI testing. Each PRI the SP sends a test word to the ECI which is then sent up to the ADC by the ECI. Two PRIs later this test word is received back at the SP from the ADC and can be used to see if there is a fault in the loop. If a fault exists, it is reported to the RDP via the SP status word. The RDP will then make the decision to set the ECI into loop-back, which will loop the ECI manchester link driver to the manchester link receiver. Data now sent by the SP will be looped back at the ECI. Checking the data will then indicate where the communication fault has occurred i.e. either in the manchester link and ADC or in the ECI and SP.

5.5 Evaluation link port

The evaluation link port will be output port 5. The output of this port will be buffered and made available to a DIN plug on the SP front panel. For diagnostics and debugging this port will be a bidirectional port. It will thus be possible to connect an IBM PC or compatible, to this port in order to send and receive data. Special debugging software routines will have to be provided to service such debugging requirements.

5.6 Watch dog timer

The SP will have a watch dog timer which will require regular resetting. The XF flag on the TMS 320 will be used to reset the watch dog. Failure to reset this timer will result in the SP being reset. The watch dog will have links to allow it to be disabled during commissioning of software. The timer must be reset within 150 ms. The manual reset switch will be connected to the watch dog circuitry.

5.7 Functional description

This functional description follows the same format as that in section three. The description in section three specifies what has to be done and this description describes how it is to done.

The function of the SP is controlled by the software and is also closely linked to the operation of the ECI. The functional description is divided up into two types of functions and several modes. The two types of functions are PRI based functions and 20 ms based functions. The modes will be defined as tracking mode, acquisition mode and setup modes. Tracking and acquisition are important because hardware functions on the ECI are utilised in certain ways while in these modes. Setup modes are not as important, since they do not require the hardware tracking functions on the ECI.

5.7.1 Setup functions

There are four phases in setup mode. In all four of these phases the SP will receive 128 words from the estimation channel each PRI. The PRI handling of this data will be exactly the same as in tracking mode. The processing, however, will be dependent on

the setup phase. The processing and detailed software design is beyond the scope of this presentation.

5.7.2 PRI based functions

5.7.2.1 Acquisition mode

a. Monitor detection status

The PRI timing will be connected to interrupt zero. On receiving this PRI interrupt the SP will monitor the ECI detection counter by reading port 1. The signal NIPECI1 (not input ECI port 1) will be asserted to enable this port. Bits 8-15 of this port indicate the detection counter value.

b. ECI samples and range register loading via dual port memory

Each PRI the dual port memory control circuitry loads the 128 words of estimation data and the 64 words from the range register FIFO, into memory. The 128 words of estimation channel data will not be used during acquisition. The range register may contain less than 64 words. When the FIFO is empty, successive reads will yield values of 0xFFFF.

5.7.2.2 Tracking mode PRI based functions

The following sequence lists, roughly in chronological order, the events that occur each PRI.

a. Read sample data from ECI

When the ECI has received all the data from the ADC, it will send a signal to the SP. Note that this

signal does not coincide with the first range bin. This is illustrated in figure 7 on page 18. This signal will start the automatic loading of the ECI data into the SP dual port memory. When the sample data and the contents of the range register have been loaded into SP memory, an interrupt 0 will be generated. On receipt of this interrupt the SP will commence PRI software based routines.

b. Write command data to ECI

The SP will write 8 words of ADC command data to port 0. The signal NOPECI0 (not output ECI port 0) will be asserted enable the write function on this port. The SP will then inform the ECI of the data by setting bit 0 of port 2. This bit will indicate to the ECI communications controller to switch the estimation serial link to transmit mode. The signal NOPECI2 (not output ECI port 2) will be asserted to enable this port.

c. Check test word

The data from the ECI comes as a packet of 128 words. The 121 'st word will be the test word which the SP sends up to the ADC each PRI. The ADC then sends this exact word straight back to the SP. If this word is corrupted then a fault in the estimation channel is indicated.

d. Read ECI status register

The SP will then read the ECI status via port 1. Signal NIPECI0 (not input ECI port 0) will be asserted activate this port. Bits 0 and 1 represent the target-in-gate and target-in-window signals and positive logic will be used i.e. low = signal not

present and high = target in gate/window. These will be passed on to the RDP. Bits 8 to 15 are the detection counter output.

e. Reset the ECI

The SP will then reset the ECI by setting bit 1 of port 2. This will reset all the PRI based hardware on the ECI such as flip flops and counters. Bits 0 and 1 of port 2 can now be reset since both outputs are pulse outputs.

f. Processing

The SP now continues to process the data from memory. Part of the monopulse and split gate calculations are done and the results are added to the total running total for each calculation in memory. If the evaluation link is active then the results for each PRI will be output to port 5. Processing does not make use of any hardware except for memory accessing.

g. Idle period

The SP will complete the processing before the next PRI sync pulse and will idle until such time as it receives a PRI sync pulse (interrupt 0) or an interrupt 1 signal which indicates the end of the 20 ms period. These interrupts are only enabled during this idle period. If a pulse interrupt occurs when the interrupts are disabled the interrupt is latched by an internal flip flop in the TMS 320C25. When the interrupts are enabled the TMS 320C25 will then react to the latched signal immediately. This ensures that interrupts can be level or edge sensitive.

5.7.3 Functions based on the system 20 ms pulse

5.7.3.1 Acquisition mode

a. Commencing acquisition mode

When the SP receives the command to enter acquisition mode via port 3, it will immediately clear all its FIFOs. This is done by toggling bit 3 of port 2. It will also reset its internal PRI counter.

b. Write data to RDP

After having been in acquisition mode for the last 20 ms the SP will send data to the RDP. This data will be written into FIFOs connected to port 3. The signal NOPMB3 (not output multibus port 3) will be asserted to enable this port. Once the data has been written into the FIFO, the RDP can read the data out of the FIFO by accessing the I/O space addressing. The SP will also update the status word by writing to port 4, and signal NOMB4 (not output multibus 4) will be asserted to enable this. The status word will be latched by a register which will be accessible to the RDP via the multibus interconnect space.

c. RDP timing

On receiving the 20 ms interrupt the SP will prepare the data to be sent to the RDP. The preparation of this data and the time delay in writing this data to the FIFOs, as described in b above, will take no longer than 660 micro seconds. During this time the RDP will send the command word and the evaluation data to the SP. By the time this has been done, the SP should have the data available for the RDP.

5.7.3.2 Tracking mode

a. Timing

Entering or leaving tracking mode will occur at the start of a 20 ms period. On receipt of the 20 ms interrupt (interrupt 1), the SP will cease to perform the usual PRI based functions. The PRI interrupt (interrupt 0) must remain enabled, and as this interrupt is received, the SP will execute a limited set of PRI based functions. These include reading in data, equalisation and the upkeep of the MTI history. On completion of these, the TMS320 will return to servicing the 20 ms interrupt.

b. Data from the RDP

While the SP is preparing the data to sent to the RDP, the RDP can update the SP command word. The RDP can also send the 20 ms data to the SP via the port 4 FIFO. During this time the SP will proceed with preparing the data for the RDP. The SP will set the busy flag, bit 3 of port 4. When the data has been prepared and has been sent to the output port 3 FIFO, the SP will clear the busy flag. This operation will take no longer than 660 micro seconds.

c. Data to the RDP

To determine when the data from the SP will be ready, the RDP can either wait a fixed time of 660 micro seconds, or it can poll the SP busy flag.

5.8 Circuit description

5.8.1 Processor

a. Clock

The TMS 320C25 clock input will be driven by a TXO400 TTL 40 MHz oscillator.

b. Reset pin

Provision has been made for a reset to be given from the RDP via the multibus. The signal NMBRST is asserted to initiate a reset. A reset may also be given by the watch dog timer, a DS1232, if the timer is not toggled at least every 150 ms. A manual push button will be connected to the DS1232 to allow manual resets. Provision is made for the watch dog to be disconnected by rearranging the links P13 and P24.

c. Interrupts

INT0 and INT1 are connected to the PRI and 20 ms signal respectively. Interrupt 0 is generated by the dual port RAM circuitry. When the ECI has received all 128 words of estimation channel data, it will send a signal to the SP. The signal "PRI" on pin 62 of the P2 connector will be asserted to indicate this. The EP600 EPLD will receive this signal and will commence loading in 128 words from the ECI. The signal NRFF, which will be connected to the READ pin on the ECI FIFO, is asserted for each read cycle. The EP600 will be programmed to read exactly 128 words using the NRFF signal, and exactly 64 words using the NRRR signal. After this an interrupt will be generated on the INT0 line.

To provide for the possibility of using a third interrupt, INT2 is connected high via a link. If this interrupt is required the link can be changed. The signal connected to the other side of the link is NLATCHN which is asserted if the SP command word is accessed by the RDP via the multibus. It is not intended to use this interrupt unless this is found to be necessary during commissioning of the radar.

The quality of the system 20 ms pulse is unknown. For this reason a 74LS123 timer is provided to ensure that the signal has a reasonably constant pulse width and is free of excess noise. Links have been provided to circumvent this timer if required.

d. BIO pin

The BIO pin on the TMS 320 is a potentially useful input flag. This pin is not used but provision has been made for a modification, by tying this pin high via its own pull-up resistor. If the pin is later required the resistor can be removed to give access to the pin.

e. The READY pin

The READY pin on the TMS 320 can be used for automatic one-wait-state operation if connected to the MSC pin. The TMS should run in zero-wait-state operation. If a timing problem occurs with an I/P port then that port can be run on one-wait-state operation by gating the port control signal with the MSC output, and connecting this to the READY pin. To provide access to the READY pin, the pin is tied high with its own pull-up resistor.

f. The RUNNING indicator

Provision has been made for an LED indicator on the front panel of the PCB. This LED will light only when the XF flag, which is used to reset the watch dog, is toggled repeatedly.

g. The TMSCLK bit on the SP status word

The TMSCLK bit is bit 0 of the SP status word. This bit is derived from the CLOCKOUT2 pin on the TMS 320. The signal on this pin is a clock signal four times slower than the input clock i.e. 10 MHz. This signal is used to retrigger a 74LS123 timer, thus never allowing it to time out. The output of the timer, TMSCLK, is thus an indication of when the TMS 320 stops running.

h. I/O and memory data bus

The TMS data bus has been buffered with different buffers for the I/O bus and the memory bus. The I/O bus is labelled IOD0-IOD15 and the memory data bus is labelled D0-D15.

5.8.2 Ports and port control

a. Ports

Seven input ports and seven output ports have been provided for. Table 2 shows all fourteen ports.

PORT	I/O	FUNCTION
0	IN	Spare 16 bit input port
1	IN	ECI status word ECI
2	IN	Spare 16 bit input port
3	IN	Command word from RDP via multibus
4	IN	Data from RDP via multibus
5	IN	Data input from the evaluation link
6	IN	SP FIFO status monitoring word
0	OUT	ADC command word to ECI
1	OUT	Range gate register to ECI
2	OUT	Interface control port (acknowledge and reset)
3	OUT	SP data output to RDP via multibus
4	OUT	SP status word to RDP via multibus
5	OUT	Evaluation link data output port
6	OUT	Range register read inhibit during 20 ms int.

Table 2. Port functions.

b. Port control

All port control is achieved by the use of the 74F138, a 3 to 8 line decoder. All port control lines are active low except output port 1, 2 and 4 which thus require inverters after the 74F138. Ports are accessed by the TMS 320 when the NIS (not I/O space) signal is asserted. The RNW (read not write) signal indicates whether the access is a read or write.

c. Port types

Ports that do not require data to be latched or written into a FIFO make use of a 74F541 octal buffer. The peripheral equipment that will connect to these ports must have latching facilities of their own. Ports that require the data to be latched make use of a 74F573 octal latch.

Input port 3 and output port 4 both use IDT2702, 1k word FIFOs. The FIFOs fulfil latching requirements of the ports and also provide for asynchronous destination or

source equipment. Port Equipment that is connected to port 5 must fulfil the timing requirements of the TMS 320 i.e. for zero-wait-state transfer the access time must be less than 40 ns. The FIFOs can be reset by toggling bit 3 of output port 2. When asserted this bit will initiate a reset of all the FIFOs i.e. for port 3 and 4. The IDT2702 FIFOs always require to be reset at power up.

Input port 6 is the FIFO monitoring port. The data transfer through the FIFOs must be carefully planned so as not to build up a residue of unused data. This can be monitored using port 6 which shows the empty and full flags. This, however, can waste critical time. It is thus probably best to use the port 6 monitoring facility for debugging problems which may arise during commissioning of the radar.

5.8.3 Memory

a. ROM

The ROMs used are the Cypress CY7C291 2 kbyte by 8 bits EPROM. The processor is not sensitive to the type of memory used as long as the access time is 35 ns or less. Provision has been made for 12 kbytes of ROM. The SP software resides in the ROM.

b. RAM

Provision has been made for 16 kbyte RAM with the use of the Intel 51C98 16 kbyte by 4 bit SRAM. As in the case of the ROM the only critical aspect of the RAM is that it should have an access time of 35 ns or less.

c. Dual port RAM

The first 16 kbytes of RAM are ordinary static RAM, but accesses to the next 1 kbyte above 16 k, will access 1 kbyte of dual port RAM. The SP will have access to the left port of this RAM, and the automatic loading circuitry to the right hand port. The first 128 locations of this memory will always contain the 128 words of estimation channel data. The next 64 words will always contain the 64 words from the range register FIFO. An EPLD device, the 56C060, is used to control the automatic transfer of data. The programming of the EPLD is fairly complicated and to ease the task an assembler, CUPL, was used to compile the fuse map. The source file for the CUPL assembler is shown below.

```
/*************************************************/
/* SOURCE CODE FOR EPLD, U18                     */
/*************************************************/

Name      DUAL3;
Partno     ;
Date      22/6/90;
Revision  01;
Designer  ;
Company   ESD;
Assembly  None;
Location  None;
Device    ep600;

 /*************************************************/
/* Allowable Target Device Types :  EP600        */
/*************************************************/

/**  Inputs  **/

Pin 1    = clock1;        /* Counter Clock 1              */
Pin 13   = clock2;        /* Counter Clock 2              */
Pin 11   = MRXD;          /* Start comnd ECI rising edge  */
Pin 2    = 10MHz;         /* 10 MHz signal                */
Pin 14   = PRFINHIB;      /* NINT0 pulse after 128 or 192 */

/**  Outputs  **/

Pin [3..10,15] = [q0..8];       /* Counter Outputs      */
Pin 16 = !Q9;                   /* q9 and count enable  */
Pin 17 = !RFF;                  /* Read ECI fifo        */
```

```
Pin 18 = !RRR;                    /* Read ECI range reg    */
Pin 19 = !OE;                     /* Output enable         */
Pin 20 = !RW;                     /* Read Write line       */
Pin 21 = !CE;                  /* Dual port RAM chip enable */
Pin 22 = !NINT0;                  /* TMS int0 line */

/** Declarations and Intermediate Variable Definitions **/

Field counter = [q8..0];   /* Declared Counter Field      */

/** Logic Equations **/

counter.t = 'd'1                          /* BIT 0 (LSB)  */
          # 'd'2    & (q0) & Q9          /* BIT 1  */
          # 'd'4    & ([q0..1]:&)        /* BIT 2  */
          # 'd'8    & ([q0..2]:&)        /* BIT 3  */
          # 'd'16   & ([q0..3]:&)        /* BIT 4  */
          # 'd'32   & ([q0..4]:&)        /* BIT 5  */
          # 'd'64   & ([q0..5]:&)        /* BIT 6  */
          # 'd'128  & ([q0..6]:&)        /* BIT 7  */
          # 'd'256  & ([q0..7]:&);       /* BIT 8  */

counter.ar = !MRXD;                       /* Reset counter */

RFF       = 10MHz & !q7 & !q8 &  MRXD;
RRR       = 10MHz & !q8 & q7 & !q6 & MRXD & !PRFINHIB;
OE        = 'b'0;
NINT0     = PRFINHIB & q7 & !q6
           # !PRFINHIB & q7 & q6;
RW        = 10MHz & !q7 & !q8 & MRXD
           # 10MHz & !q8 & q7 & !q6 & MRXD & !PRFINHIB;
CE        = 10MHz & !q7 & !q8 & MRXD
           # 10MHz & !q8 & q7 & !q6 & MRXD & !PRFINHIB;
Q9.d      = MRXD & !q8;

 /*************************************************************/
```

5.8.4 Multibus interface

5.8.4.1 General

The SP occupies 2 kbytes in both the I/O and the interconnect space of the multibus. Access to an agent via the interconnect space is achieved by using the slot number of that agent as an address. At power up the multibus central services module allocates a slot number to each agent on the bus. Access to an agent via I/O space is achieved by using a base address for each agent as the agent's

address. The base address register is assigned to each agent at power up. The interface makes use of two PALs, U15 and U68 (on sheet 4 of the circuit diagrams), for timing and function decoding, respectively.

5.8.4.2 Timing decoding

The source file for the CUPL assembler is shown below. The following meanings apply:

& - AND function
! - signal invert
- OR function
.d - synchronous output
.OE - output enable

```
/*************************************************************/
/* CUPL source code for the multibus timing decoder          */
/*************************************************************/

Name      MB2TIM;
Partno    ;
Date      08/08/90;
Revision  01;
Designer  Peter van der Linden;
Company   ESD;
Assembly  None;
Location  None;
Device    GAL16v8;


/*************************************************************/
/* Allowable Target Device Types :  GAL16v8, PAL16R8         */
/*************************************************************/

/**  Inputs  **/

Pin 1  = NCLK;
Pin 2  = !NSC2;
Pin 3  = !NSC3;
Pin 4  = !NSC0;
Pin 5  = IREAD;
```

```
Pin 6  = !LNSC2;
Pin 7  = !LNSC3;
Pin 8  = !NIOVALID;
Pin 9  = !NICVALID;
Pin 11 = !OE;

/**  Outputs  **/

Pin 12 = !ISC7;
Pin 13 = !ISC6;
Pin 14 = !NICSEL;
Pin 15 = !NIOSEL;
Pin 17 = !NBUFFEN;
Pin 18 = !ISC4;
Pin 19 = !ISC5;


/** Declarations and Intermediate Variable Definitions **/
/** None                                                **/

/** Logic Equations **/

NBUFFEN.d  = !NSC0 & NSC3 & !NBUFFEN & (NIOVALID # NICVALID);
NICSEL.d   = !NSC0 & NSC3 & NICVALID & !NICSEL;
NIOSEL.d   = !NSC0 & NSC3 & NIOVALID & !NIOSEL;
ISC4       = !NSC0 & NSC3 & (NIOVALID # NICVALID);
ISC5       = (NIOVALID & (NSC2&NSC3 # !NSC2&NSC3
             # !NSC2&!NSC3))
           # (NICVALID & (NSC2&NSC3 # NSC2&!NSC3
             # !NSC2&!NSC3));
ISC6       = !NSC2;
ISC7       = IREAD & NIOVALID;
NBUFFEN.OE = 'B'1;
NICSEL.OE  = 'B'1;
NIOSEL.OE  = 'B'1;
/*************************************************************/
```

5.8.4.3 Function decoding

a. Slotcode

When the signals ILATCHN, NRST and NARB5 are asserted, the signal SLOTCOD is asserted. This enables a 74F377, which latches the slotcode presented on lines ARBO to ARB4, with the first rising edge of NBCLK.

b. I/O and IC select

When the multibus signal SC0 is low, the request phase of the multibus transfer is indicated and latches U26 and U23 are enabled. With the first rising edge of NBCLK, the multibus signals AD9 to AD14 and SC2 to SC5 become valid. These signals are presented to two comparators U35 and U36 (on sheet 4 of the circuit diagrams). U35 indicates that an I/O space access is in progress, and U36 indicates that the interconnect space is being accessed. These signal are routed through the timing PAL so that NIOSEL and NICSEL become synchronous with the reply phase of the transfer. These two signals can now be used with the NAD9 and NAD10 signals to decode all the various transfers required.

c. Function decoding PAL

The source file for the function decoder is shown below. This source file is to be assembled using PALASM2. The following meanings apply:

. - AND function
/ - signal invert
+ - OR function

```
;*************************************************************
; PALASM2 source file for the multibus function decoder
;*************************************************************

Title     Multibus II function decoder
Pattern   MB2FNCN.pds
Revision  A
Author    Peter van der Linden
Company   ESD South
Date      8 Auggust 1990

CHIP MB2FNCN PAL20L10

;The function decoder uses a PAL 20 L10. This source file is
;to be assembled using PALASM2
```

```
;  Pin Assignments

; Inputs

;  1        2        3      4      5        6       7       8    9
/NIOSEL /NICSEL /NAD10 /NAD9 IREAD /ILATCHN /NRST /NARB5 NC
;10 11 12
 NC NC GND

; Outputs

;13    14         15         16          17         18         19
NC /NADSEL /NIOBASE /NSLOTCOD /NSTATUS /NMB2RST /NCARDID
; 20          21         22        23    24
/NCMDSEL /NLATCHN /NFIFOW /NFIFOR VCC


EQUATIONS

NADSEL   = NICSEL * /NAD10 * NAD9 * IREAD
NFIFOR   = NIOSEL * /NAD10 * NAD9 * IREAD
NFIFOW   = NIOSEL * /NAD10 * NAD9 * /IREAD
NLATCHN  = NICSEL * /NAD10 * /NAD9 * /IREAD
NCMDSEL  = NICSEL * /NAD10 * /NAD9 * IREAD
NCARDID  = NICSEL * NAD10 * NAD9 * IREAD
NMB2RST  = NICSEL * NAD10 * /NAD9 * /IREAD
NSTATUS  = NIOSEL * /NAD10 * /NAD9 * IREAD
NSLOTCOD = ILATCHN * NARB5 * NRST
NIOBASE  = NICSEL * /NAD10 * NAD9 * /IREAD

;************************************************************
```

5.9 Circuit diagram

The final circuit diagram is presented in a set of four sheets. These sheets have been included in a sleeve, at the back of this report.

6 Preliminary testing of the completed circuit

6.1 Manufacture and commissioning

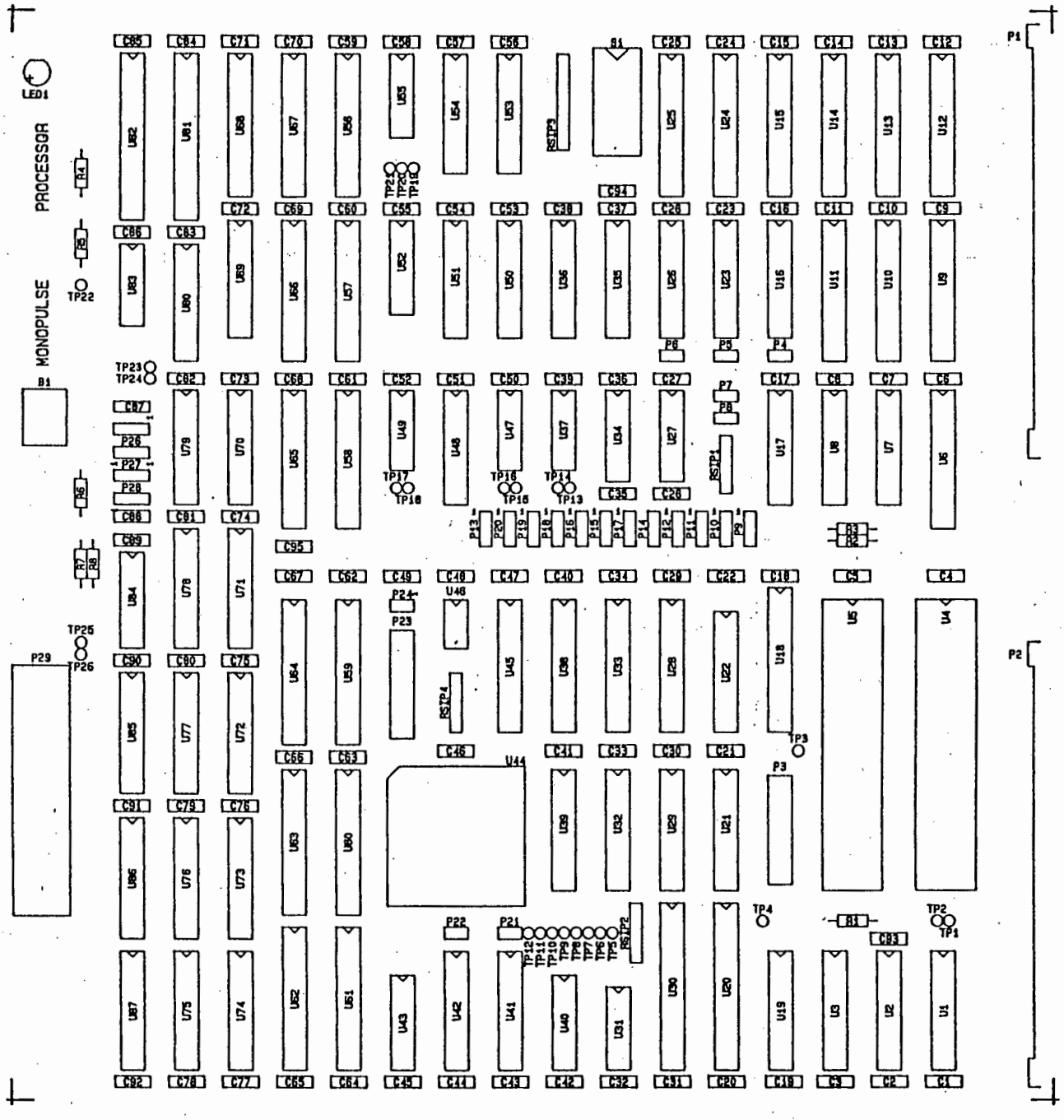
After the final schematics were completed, the circuit was sent away for manufacture. The board is a six layer board and the PCB layout and component side artwork is shown overleaf.

Under normal operation the circuit draws 3.2 amps and generates a considerable amount of heat. The final housing for the radar computer system will therefore require forced air cooling.

6.2 Processing time tests

The ability of the processor to complete all the processing within the required time is the most important factor in determining the success of the design. Detailed tests were done in order to verify the total time taken during track mode, which is the most time consuming mode.

The time tests performed using the prototype circuit were done by loading each routine to be tested separately. The routine was then run a million times and the time measured in seconds was equivalent to the actual time in micro seconds. This method however, excludes the time taken for the processor to branch to, and return from, the routine. In the case of interrupts, it excludes the processors response time to the interrupt. It also excludes the time taken for the software to step through the main routine which calls the various subroutines.



ISSUE	A				APPROVED
DATE	04/07/90				
ECN NO	------				DRAWN
APROVED					GMB
DWG NO :					SHEET
TITLE :	MONOPULSE PROCESSOR COMPONENT SIDE IDENT ARTWORK				3 OF 12

Figure 10. Component side artwork of the processor

These excluded items, however, only constitute a small percentage of the total time taken to run the entire program.

The time tests shown in table 3 below were done running the entire program in track mode. To measure each routine, a flag was set just before the call to that routine, and reset after returning. An oscilloscope was used to measure the exact time taken. Although no further speed optimisation has been implemented in the software since the prototype testing, extensive software debugging and testing has been done.

FUNCTION	TIME micro s
a. Sample combine	26
b. Dynamic equalisation	60
c. MTI filtering	85
d. Update running PRI based history	33
e. Split gate calculation	54
f. Monopulse calculation	10
g. PRI select	5
h. AGC, response filter update	2
i. Read ECI status	8
j. Write ECI data	8
k. Evaluation data out	20
TOTAL	311

Table 3. Process execution time measurements

The total time available is 333 micro seconds. Therefore, there is a spare time of approximately 20 micro seconds. By mapping the external RAM data for tracking to internal RAM, there is an additional time saving of 20 micro seconds (the TMS320 C25 requires an extra cycle to run certain instructions using external memory). The most time consuming routine is MTI filtering. This routine is written entirely in C,

therefore there is scope for substantial (approximately 20 micro secs) improvement in execution time.

The software has been arranged so that when a 20 ms interrupt is received, the monopulse calculation is suspended while the processor collates data. Measurement of this 20 ms handling routine shows that it takes 140 micro seconds. This is far less than one PRI, thus no more than one monopulse sample per 20 ms would be lost. This is a loss of one sample out of 54 each 20 ms. However, the spare 40 micro seconds could be used to run the 20 ms routine over 4 PRIs. If the software is modified to do this then no samples need be discarded at all.

7 Conclusion

The main objective of this project was to design a signal processor using a TMS320 C25, that can be used for monopulse radar. Some aspects which have been fundamental to the design of the SP are discussed below.

7.1 The use of the TMS320 C25

The TMS320 C25 can be used for this type of application, but it is not the ideal processor to use, for the following reasons:

The TMS320 C25 is intended for DSP and therefore has special features dedicated to DSP work. Some of these features are bit reversed addressing for FFT calculations and multiply, add and automatic address increment in a single instruction, for FIR filter design. The software required for a monopulse radar (excluding pulse doppler radar) does not use any of these features.

The TMS320 C25 is advertised as being a 40 MHz processor. In practical terms this can be misleading since the cycle time is based on a quarter that frequency i.e. 100 ns. While this is still very fast, it is not the fastest by modern standards. Processors with cycle times of 40 ns are now easily obtainable.

A more acceptable processor to use would be one that supports floating point arithmetic, that has good I/O facilities and that has a fast enough cycle time.

If, for example, the software was split into three sections, I/O, split gate and monopulse calculations,

the three blocks could run in parallel. Processors which support parallel processing, such as transputers, could be used to execute the software concurrently. Alternatively, if sequential execution is preferred, newer RISC (reduced instruction set computer) processors could be used, such as the TMS320 C30. The TMS320 C30 supports full floating point arithmetic and has a cycle time of 60 ns. Both the C25 and the C30 use Harvard Architecture.

It is probably true to say that the software will generally expand to fill the available execution time.

7.2 PRI data quantity

The quantity of data transferred each PRI, was initially underestimated. Taking into account all I/O operations during tracking mode about 200, 16 bit words are transferred each PRI. This results in the processor spending approximately a third of the available time doing I/O.

The "IN" and "OUT" instructions on the TMS320 C25 are specified as taking two and one cycle respectively. It was thus originally estimated that 200 words of I/O could take a maximum of 400 cycles. Tests have shown that, on average, five cycles are needed for each I/O. The five cycles takes into account the software overhead in setting up the stack, which is used for looping control etc. Five cycles times 200 I/O instructions takes 100 micro seconds, which is almost a third of the available time.

7.3 C Compiler

The C compiler, while inefficient, is capable of generating code that can be executed in the available time.

The disadvantage of the C compiler is the loss of processing speed. This occurs because the compiler builds a comprehensive run-time stack which takes time to maintain. It also builds a more complicated structure of assembly code, in some cases, than is possible by manually written code. To create faster code, the entire software package could be rewritten in assembler. Preliminary tests suggest a possible saving of approximately 20 %.

The advantages of using the C compiler far outweigh the loss of time. Using a higher level language allows quicker development of software. It is also much easier to maintain than assembler, which is always difficult, even for someone familiar with the code. The use of C allows a better program structure to be created. This is, however, sometimes a disadvantage to speed because looping takes longer than repeated blocks of code.

7.4 Complexity of the software

The completion of the project has resulted in a clear understanding of the requirements of a signal processor in a monopulse radar. Two comments can be made with regard to the complexity of the software. Firstly, the algorithms used and mathematics required are relatively simple. These sections of code must be written to enhance speed, not program structure. The size of the final code for this SP is be about seven kbytes. Secondly, the timing of processes within the SP are critical. The PRI and 20 ms interrupts are totally

asynchronous. Also, the response of the SP to these interrupts is mode dependent. This makes debugging very difficult.

Because there are so many I/O operations to various ports each PRI, stand-alone testing of the board remains a problem. It is impossible to test the software completely without an ECI or RDP. Test software has been written to try to simulate various conditions. Although test software is not part of the process software, it does contribute to the overall complexity of software development.

References

1. Introduction to Radar Systems. Merrill I. Skolnik McGraw-Hill 1980.
2. Introduction to Airborne Radar. George W. Stimson Hughes Aircraft Company 1983.
3. Radar System Analysis. David K. Barton Prentice Hall 1964.
4. Radar System Design and Analysis. S A Hovanessian Artech House 1984.
5. Monopulse Priciples and Techniques. S M Sherman Artech House 1984.
6. Multibus II bus architecture specification. Intel Corporation 1983.
7. Texas Instruments TMS320 C25 Users Guide. Texas Instruments 1988.

APPENDIX A - Design requirements

A signal processor shall be designed to function in a monopulse radar environment and shall fulfil the following requirements:

1 PRF range

The processor shall cater for low PRF radars of up to 3 kHz.

2 Function requirements

The processor shall perform all software functions which require to be done on estimation channel data in a monopulse radar at PRF speed. Functions which can be done at a slower rate will be done in some other processor and the signal processor shall format it's output accordingly. The high speed functions shall include at least the following:

i) Receiver channel error compensation.
ii) MTI filtering. The MTI filter shall be a triple pulse canceller.
iii) Angle error calculation (monopulse calculation).
iv) Range error calculation (split gate calculation).
v) AGC of receiver using STC facilities. The AGC control output shall be damped with a first order filter. The processor shall store a set of STC curve data in memory which can be used for the AGC. Envelope detection or a similar method will be used to decide if the receiver gain should be increased or decreased. The decision will be taken to increase or decrease the gain (select a different STC curve) if the envelope is greater or smaller than predetermined factors.

vi) Other functions as found necessary eg. Scaling of data, averaging data etc.

3 Interface requirements

i) The processor shall receive monopulse estimation data in the form of samples of the sum channel (I and Q), the azimuth channel (I and Q) and the elevation channel (I and Q). This shall make a total of 120 samples, each of which is a 16 bit word. This data shall be read in and processed every PRI.

ii) Data shall be ready for output every 20 ms. Every 20 ms the processed data shall be read by another processor, in an asynchronous read operation. Between each 20 ms read, the processor should integrate the monopulse results for each PRI thus improving the SNR. How the data should be integrated is to be determined.

iii) The processor shall receive all its high speed data via an interface card, the ECI (estimation channel interface). The processor shall also read and write each PRI to the detection counter, the range register and the range gate register which are all located on the ECI. (see glossary for definitions of the detection counter and the range and range gate registers).

iv) The processor shall communicate to other processors via the Multibus II iPSB parallel bus system. The signal processor shall not be a master on this system and thus will only receive data or make data available to be read. It will not be able to arbitrate for bus ownership, and must treat all communications over the Multibus as asynchronous. The Multibus II can communicate on the parallel bus using four different address spaces; memory space, message passing, I/O space and interconnect space. Of these

the processor must have the ability to use the I/O space and the interconnect space.

v) The processor shall output data to an output port, called the evaluation port, such data as required to evaluate all the software processes in the processor during normal operation. The nature and format of this data is to be determined.

4 Synchronisation

The processor shall have two main timing inputs, a PRI sync pulse and the 20 ms read pulse. The signal processor shall derive all its synchronisation from these inputs and interrupts shall be implemented accordingly.

5 Control

The processor shall receive mode control instructions via the Multibus. The required modes are to be determined. The processor shall incorporate as many self test features as is practical and necessary.

6 Status

The processor shall report its current status via the Multibus. The format of this status is to be determined.

APPENDIX B - Data structure definition

All inputs and outputs to the SP shall be TTL compatible. Outputs from the SP shall not be loaded by more than two Low Power Schotky gates per output. Described below are the data structures for the ECI ports, the multibus interface ports and the evaluation link.

1 ECI ports

All communication with the ECI shall occur at a PRI rate ie. every 333 micro seconds.

a. Bit structure

The bit structure of each input and output word is defined below in tables 4 and 5.

TYPE	TITLE	BITS	BIT FUNCTION
A	Estimation sample data port	0-15	16 bit word
B	ECI status word	0 1 2 3 4 5 6 7 8-15	Target in gate Target in window Message sent to ADCU Range register empty Manchester link active Detect count overflow ECI to ADCU fifo full ADCU to ECI fifo empty Detection counter
C	Range register port	0-16	Contents of range reg.

Table 4. Bit structure of words input from ECI

TYPE	TITLE	BITS	BIT FUNCTION
D	ADCU command port	0-15	16 bit words
E	Range gate register	0-16	Range gate reg content
F	Interface control	0 1 2 3 4-15	Message received Acknowledge to ECI ECI reset SP external reset Not used

Table 5. Bit structure of words output to ECI

b. Word structure

Some ports shall require more than one access per PRI. These ports shall be the sample data, the ADCU command word and the range register. All other ports to the ECI shall input or output only one word per PRI. Where multiple access is required the word structure is shown in table 6. The TYPE in the figure refers to the bit structure outlined in tables 4 and 5. The WORD NUM refers to the word number.

TYPE	TITLE	WORD NUM	WORD FUNCTION
D	ADCU command port	0 1 2 3 4 5 6 7	Range gate position Not used Next PRF Next STC Test word Not used Not used Not used
E	Estimation sample data port	0- -120 121 122-128	SUM data AZIMUTH data ELEVATION data Test words Not used
C	Range register	0-63	Ranges at which detection are registered

Table 6. Word structure for ports to the ECI

2 Multibus II interface ports

The SP shall provide a Multibus II interface in order to transfer radar data to and from the SP and the RDP, to indicate SP status information to the RDP and for the transfer of commands from the RDP to the SP.

The multibus can communicate between agents using memory space, IO space, interconnect space and message passing. The multibus on the SP shall provide facilities for the RDP to access it via IO space and interconnect space. The SP shall occupy a maximum of 2 kbytes in the MBII I/O space and this shall be addressed using an IO base address register. The SP shall also occupy 2 kbytes space in the MBII Interconnect space. The interconnect space is accessed by using a card slot ID number.

The interface shall allow for correct card slot insertion verification. There shall thus be a set of switches on the SP which shall be set to indicate the desired slot number. The RDP can then read these switches and thus determine if the SP is in the correct slot or not. There shall also be command register read-back and I/O base address register read-back checks. When the RDP sends a command or the IO base address to the SP, the SP shall latch this data. In order to check that the correct data has been latched the RDP shall be able to read back the data by reading from the same address that the data was written to.

a. Bit structure of interconnect space ports

Table 7 shows the bit structure of words transferred via the interconnect space.

TYPE	TITLE	BITS	BIT FUNCTION
F	IO base address	0-7	8 bit word
G	SP command word	0	SP mode control A
		1	SP mode control B
		2	SP mode control C
		3	SP mode control D
		4	Spare
		5	Eval link on/off
		6	AGC on/off
		7	MTI on/off

Table 7. Bit structure for interconnect ports

b. Bit structure of IO space ports

Table 8 shows the bit structure of words transferred via IO space.

TYPE	TITLE	BITS	BIT FUNCTION
H	SP data output	0-15	16 bit words
I	SP status word	0	TMS clock active
		1	PRI sync active
		2	Not used
		3	SP busy
		4	Error code A
		5	Error code B
		6	Error code C
		7	Error code D
		8	Error code E
		9	Not used
		10	Not used
		11	Target in window
		12	Target in gate
		13	Detect count overflow
		14-15	Not used
J	RDP data input	0-15	16 bit words

Table 8. Bit structure for IO space ports

c. Word structure

All data transfer to and from the RDP shall occur at 20 ms intervals. Some ports shall send and receive a block of words. Ports that require such multiple transfer are the SP data output and RDP data input. The data flow through these ports is mode dependent. Table 9 shows the word structure for these ports according to mode.

TYPE	TITLE	WORD NUM	WORD FUNCTION
H	SP data output Setup mode DC offset adjust	0 1 2 3 4 5	Sum I dc offset Sum Q dc offset El I dc offset El Q dc offset Az I dc offset Az Q dc offset
J	RDP data input Setup mode DC offset adjust	NONE	

Table 9a. Multibus ports during setup mode

TYPE	TITLE	WORD NUM	WORD FUNCTION
H	SP data output Acquisition mode	0 1 2-3	Targ start PRI num Targ stop PRI num Average range pos
J	RDP data input Tracking mode	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19	PRI pattern number STC curve number Range gate position Evaluation data " " " " " " " " " " " " " " " " "

Table 9b. Multibus ports during acquisition

TYPE	TITLE	WORD NUM	WORD FUNCTION
H	SP data output Tracking mode	0-1 2-3 4-5 6-7 8-9 10-11 12 13 .. .. 26 27	Range numerator Range denominator El numerator El denominator Az numerator Az denominator Detection data " " " " "
J	RDP data input Tracking mode	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19	PRI pattern number STC curve number Range gate position Evaluation data " " " " " " " " " " " " " " " " "

Table 9c. Multibus ports during tracking

3 Evaluation link port

The evaluation link shall be able to be activated during acquisition and tracking mode. The link shall output ten words per PRI. The selection of which ten words shall be done by the RDP. This selection is indicated in word 19, "Mode and eval select". The first 8 bits of this word shall indicate the mode status of the radar and the second 8 bits

shall instruct the SP which ten words shall be output. The bit structure of the evaluation output port is shown in table 10.

TYPE	TITLE	BITS	BIT FUNCTION
K	Evaluation output	0-15	16 bit word
L	Word 19 of RDP data input during tracking and acqu.	0-7 8-15	Evaluation words sel Radar mode status

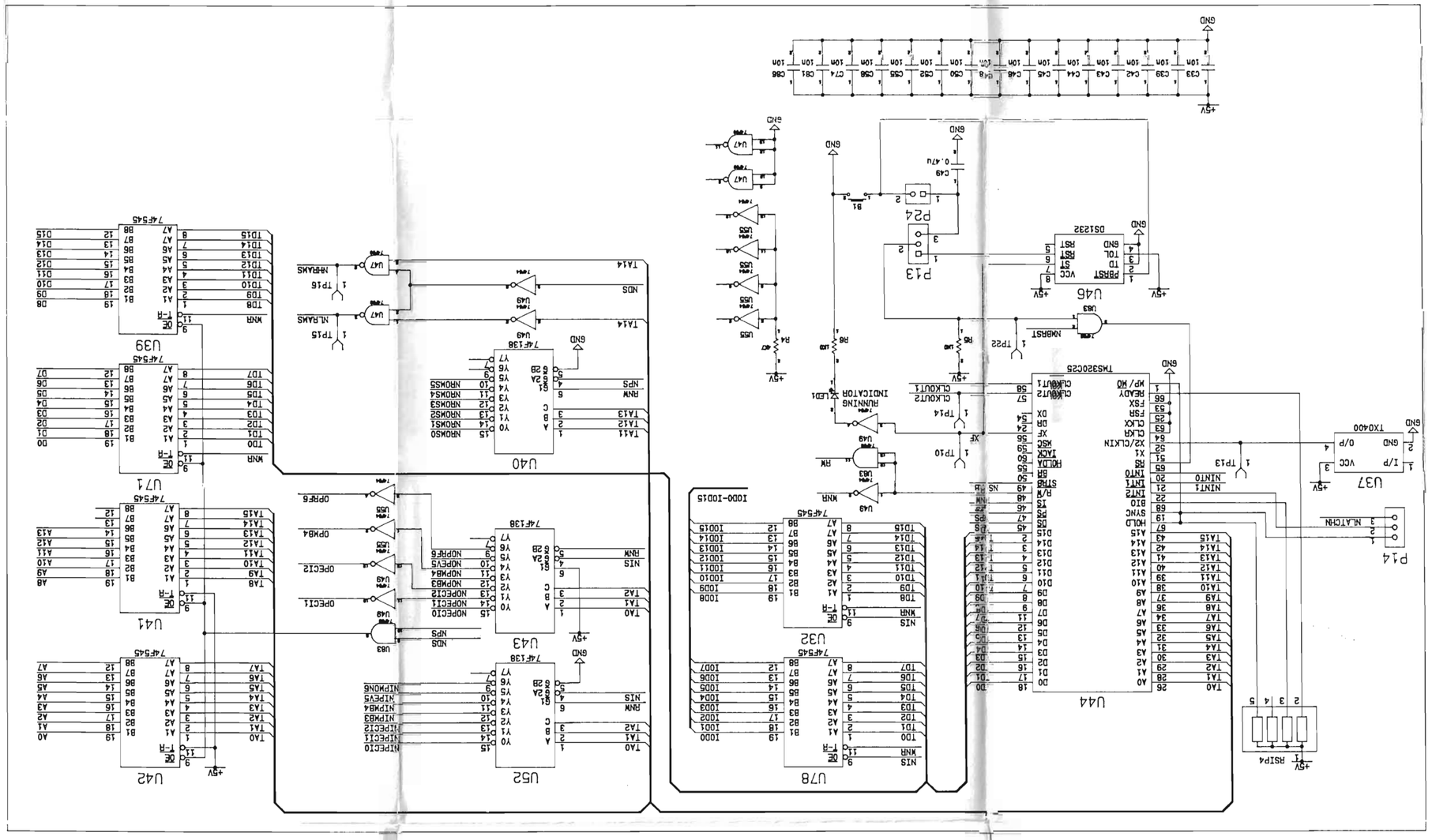
Table 10. Bit structure for the evaluation port

4 Synchronisation

The SP requires to be synchronised to two external events and shall do this by using the interrupt facility. The first event is the start of PRI pulse. This signal will be used by the SP to commence PRI based routines and to count the PRIs. It will take a certain time to process all range the bins and send the 128 words down the estimation link. When this data arrives at the ECI, the ECI shall trigger the PRI interrupt on the SP. The second event will be the RDP reading data every 20 ms and on receiving this interrupt the SP will commence the 20 ms based routines.

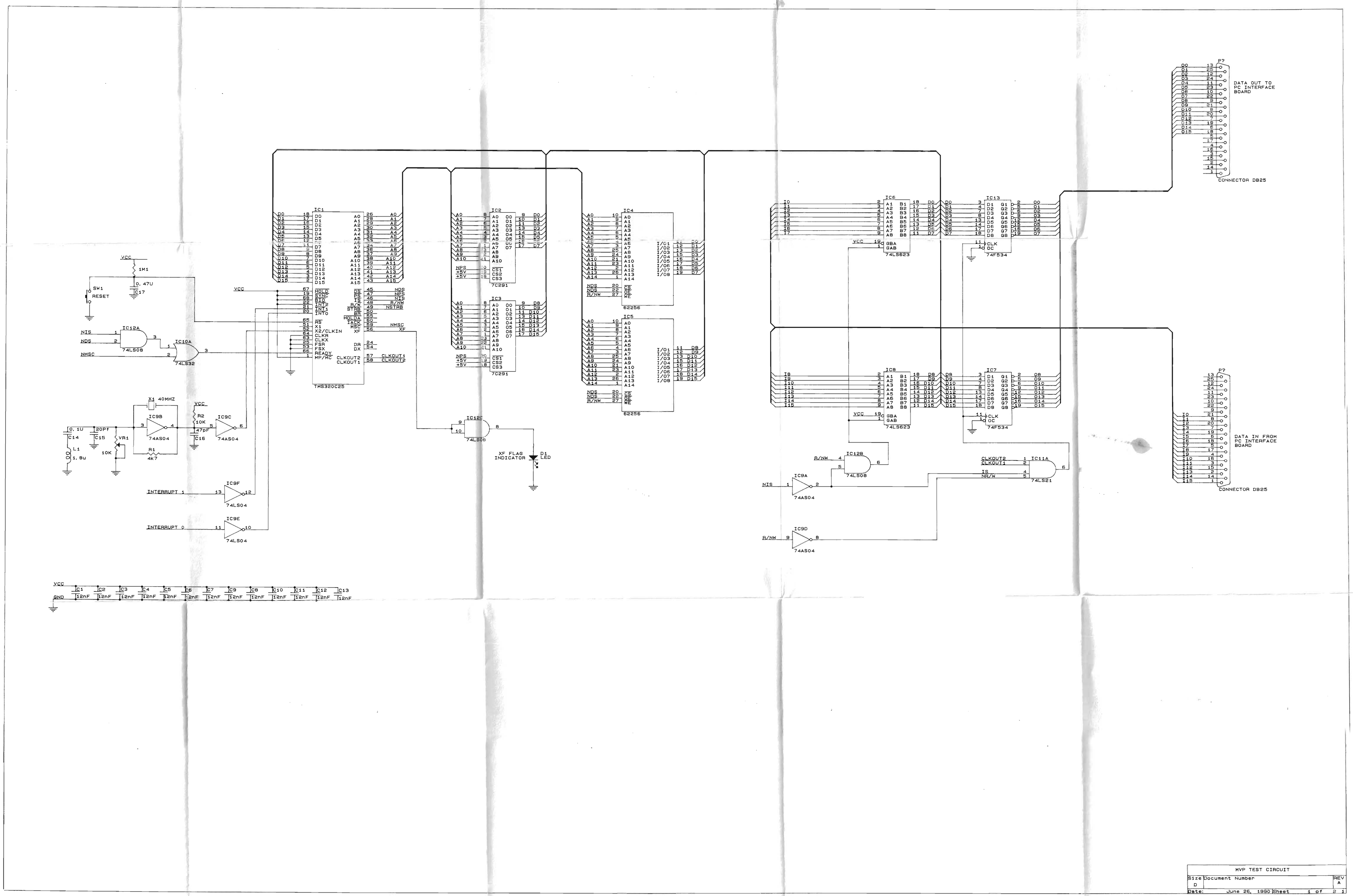
While the 20 ms routines are being executed during tracking mode the SP must still react to the PRI interrupt in order to maintain the MTI filters. Thus the PRI interrupt must have the higher priority.

SIGNAL PROCESSOR
Circuit Diagram
Sheet 1 of 4

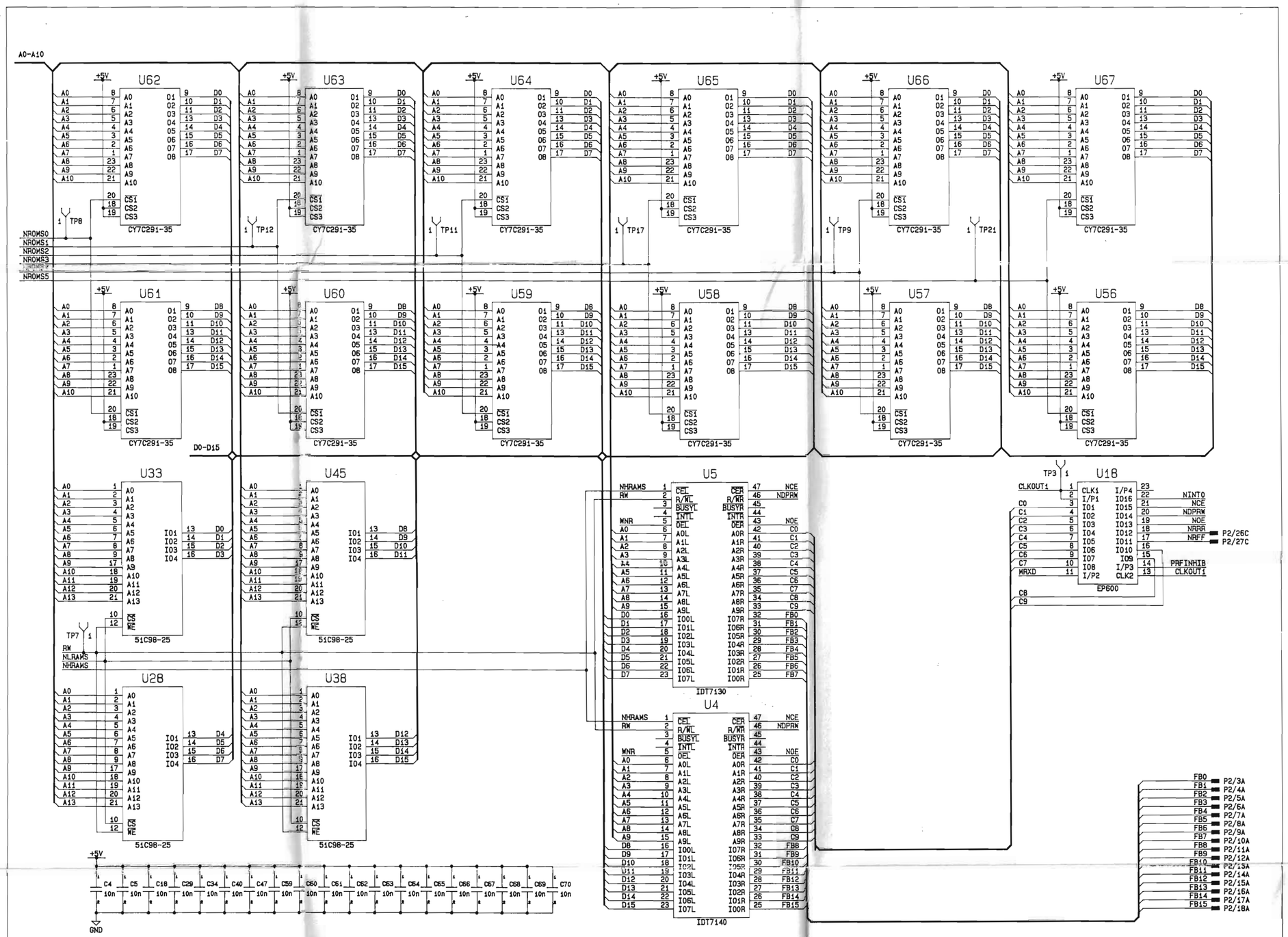


SIGNAL PROCESSOR
Circuit Diagram
Sheet 1 of 4

PROTOTYPE TEST
Circuit Diagram
Sheet 1 of 2



MVP TEST CIRCUIT		
Size D	Document Number	REV A
Date: June 26, 1990	Sheet 1 of 2	1



SIGNAL PROCESSOR
Circuit Diagram
Sheet 2 of 4

SIGNAL PROCESSOR
Circuit Diagram
Sheet 3 of 4

SIGNAL PROCESSOR
Circuit Diagram
Sheet 3 of 4

SIGNAL PROCESSOR
Circuit Diagram
Sheet 4 of 4

SIGNAL PROCESSOR
Circuit Diagram
Sheet 4 of 4

A11 NAEN AEN
B13 NIOW NW
B14 NIOR NR

DATA LINES FROM PC

ADDRESS LINES FROM PC

74F245 DIR G

PAL16L8A PAL1

PAL16L8A PAL2

NB2A
NOE
NR
NW
+5V
A0
A9
RST
NW0
NW1
NR1
NR0
NFFA

+5V
R1 470 K

U1C 74F04 NW0 R2 1K CLEAR D1 LED
U1B 74F04 NFFA R3 1K GREEN D2 LED
U1A 74F04 NEFA R4 1K RED D3 LED
U1F 74F04 NFFC R5 1K GREEN D4 LED
U1E 74F04 NEFD R6 1K RED D5 LED

A B C D IDT7202

OUT TO TMS BOARD
MVP I0 MVP I1 MVP I2 MVP I3 MVP I4 MVP I5 MVP I6 MVP I7 MVP I8 MVP I9 MVP I10 MVP I11 MVP I12 MVP I13 MVP I14 MVP I15

IN FROM TMS BOARD
MVP O0 MVP O1 MVP O2 MVP O3 MVP O4 MVP O5 MVP O6 MVP O7 MVP O8 MVP O9 MVP O10 MVP O11 MVP O12 MVP O13 MVP O14 MVP O15

MVP 623 EN FROM TMS BOARD

MVP 534 CLK FROM TMS BOARD
U1D 74F04

PC INTERFACE FOR MVPTEST PCB

Size	Document Number	REV
B		A

Date: June 26, 1990 Sheet 2 of 2