DSP-BASED VECTOR AND CURRENT CONTROLLERS FOR A PERMANENT MAGNET SYNCHRONOUS MOTOR DRIVE

BY
P. Pillay, C.R. Allen and R.Budhabhathi
Department of Electrical & Electronic Engineering
University of Newcastle upon Tyne
NE1 7RU, England

ABSTRACT

The purpose of this paper is to present the design of DSP-based vector and current controllers for a PMSM drive. Several different implementations of the vector controller are evaluated and the consequent effect on the maximum motor speed examined. A careful choice of the cosine look-up table which is used in the vector controller is presented. The possibility of implementing a hysteresis current controller algorithm using the DSP is also evaluated. The layout of a PC board including the DSP chip, latches and PAL chips to decode the I/O ports is included as well. Experimental results are provided.

1. INTRODUCTION

The drives industry is steadily moving towards digital implementation of the controllers. This can improve reliability, reduce drift associated with analog components and incorporate self diagnostics. In addition, communication between drives becomes easier for complex operations requiring more than one drive like flexible maunfacturing systems and robotics. Several approaches are possible in the development of digital drives including microprocessors, microcontrollers, transputers, ASICs, dedicated digital hardware and DSP-based systems. The DSP has several attractive features which make it suitable for high speed, real time control, including a 16 bit hardware multiplier in one machine cycle.

The analog-based permanent magnet synchronous motor (PMSM) drive has established itself in the industry for precision speed and position applications. Digital-based controllers of these drives are beginning to be reported [1-4]. In [1] the speed loop is implemented digitally with the internal current loop implemented with analog circuitry. In [2], a DSP-based vector controller is reported but very little information is provided on the implementation. In addition, the particular DSP used required external memory, which is undesirable from a pin-count, processing speed and reliability point of view.

The purpose of this paper is to present the design of DSP-based vector and current controllers for a PMSM drive. Several different implementations of the vector controller [5-8] are evaluated and the consequent effect on the maximum motor speed examined. A careful choice of the cosine look-up table which is used in the vector controller is presented. The design is centered around the TI TMS 320E17/C17 and 320E15/C15 DSPs, which are truly single chip controllers. They have 4k words of eprom/rom which are used for the vector/current controller algorithms and 256 words of RAM for temporary data storage. Scaled integer arithmetic is used througout and detailed program listings, with comments are included so that it should be possible for the reader to implement these designs. The layout of a PC board including the DSP chip, latches and PAL chips to decode the I/O ports is included as well. Experimental results are provided.

2. DESCRIPTION OF THE VECTOR CONTROLLED DRIVE SYSTEM

Figure 1 is a schematic of the vector-controlled drive system under consideration. A position servo is shown. The commanded position is compared with the actual to yield a position error. This is operated upon by the position controller to give the commanded speed. The commanded speed is compared with the actual to yield the speed error, which in turn is used by the speed contoller to form the magnitude and angle of the stator current vector. The vector controller uses the angle of the stator current vector, which ranges from 90^0 to 180^0 with respect to the rotor flux, and adds it to the instantaneous rotor angle to form the commanded phase currents. This operation is shown graphically in figure 2.

The d-axis is chosen to lie on the axis of the magnet (rotor reference frame) with the q-axis leading the d by 90 electrical degrees. The instantaneous position of the rotor is measured using for example an absolute position optical encoder or a resolver. In this particular application, it was decided to use a resolver, because of its a mechanical ruggedness and ability to reject common mode noise. The resolver output is decoded by a resolver to digital (R/D) converter to yield a 10 to 16 bit representation (depending on the resolution required by the user) of the position. A specification in angle resolution of 5.3 arc-minutes at a speed of 3000 rpm was chosen to reflect the type of precision expected in industrial drives today. The angle ϕ is located 90 degrees from θ_r, in the constant torque mode, but could approach 180^0 in the flux weakening mode of operation. The angle $\theta=\theta_r + \phi$ is the angle used in the inverse Park transformation to calculate the commanded abc currents.

The output of the vector controller are the commanded stator currents. Up to now, most of the the current control algorithms have been implemented with analog circuitry, with D/A converters being used to convert the digital commanded current into a analog value. In this paper, the possibility of implementing the current controller in DSPs is also evaluated. In this case, the commanded stator currents are then fed into a second DSP which executes the current control algorithm. It was found that one DSP could not handle both algorithms, because of the specification in the operating speed and position resolution of the machine. The above method was the logical choice for load sharing between the DSPs. The current control algorithm compares the commanded currents from the vector controller with the actual from the motor to decide on the appropriate inverter transistor to switch in order to force the actual current to track the commanded. In this design, a hysteresis current controller is implemented.

90/CH 2935-5/90/0000-0539$01.00©1990 IEEE

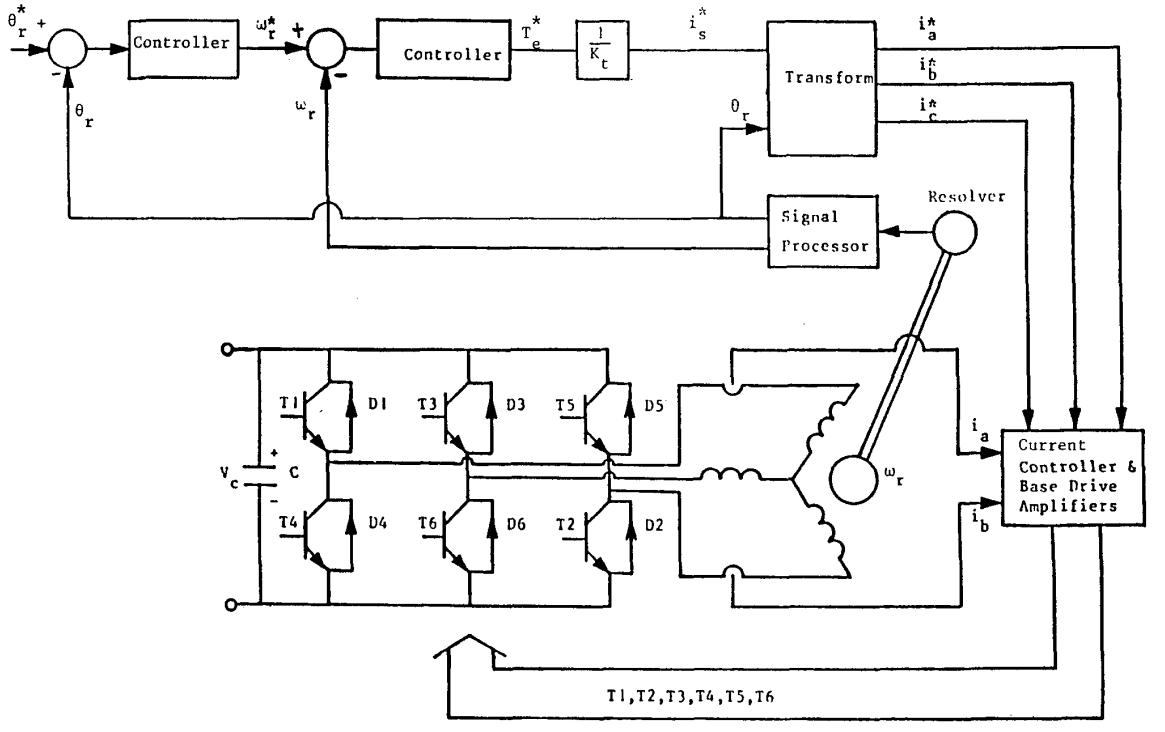

Figure 1. Schematic of a vector-controlled drive

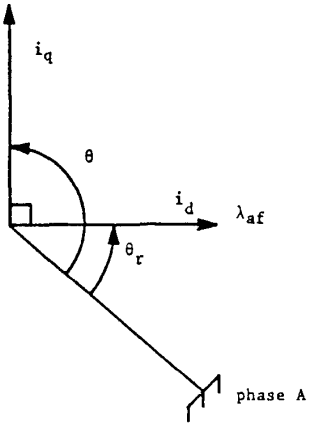

Figure 2. Vector diagram of a PMSM

3. BACKGROUND INFORMATION ON DSPs

The move towards single chip controllers in general, demanded a DSP with internal EPROM/ROM for this application. TI devices were chosen because of their availability, extensive documentation and proven ability in other areas like filtering, image processing, instrumentation and communications. The specification for internal memory narrowed the choice to the EPROM-based TMS 320E17 and TMS 320E15 (the C versions being ROM-based). In an industrial situation, the initial development could be on a EPROM-based device, with the permanent working program stored in ROM.

The TMS320E17/C17 has 6, 16 bit input/output ports, 4k words of eprom/rom (program memory) and and 256 words of RAM (data memory) for temporary data storage. Perhaps its most attractive feature is that it can perform a 16X16 bit integer multiplication in 1 machine cycle. The internal architecture consists of 1 accumulator, 2 auxiliary registers (used mainly for indirect addressing), a 4 word deep stack, a timer, dual serial port, data page and auxiliary register counters. The eprom version has a 200 ns cycle time while the rom version is 100 ns.

The TMS320E15/C15 has a similar architecture to the above except that it has 8, 16-bit input/output ports instead of the timer and dual serial ports. Both the E and C versions have a 200 ns cycle time though the TMS320E15-25/C15-25 have 160 ns cycle times.

The TMS320 series of devices have a few disadvantages. Firstly, they cannot write to an output port directly from program memory, or load data into the accumulator directly from program memory. The data must first be transferred to data memory which increases the program cycle time. The fact that the eprom and RAM are internal means that the ports cannot be memory mapped to reduce the read/write time. In addition, although the hardware multiplier can operate in one machine cycle, the multiplier must be in RAM, with the multiplicand in the T register. The 32 bit product goes into the P register from where it can only be moved to the accumulator. This increases the actual number of instructions to complete the multiplication.

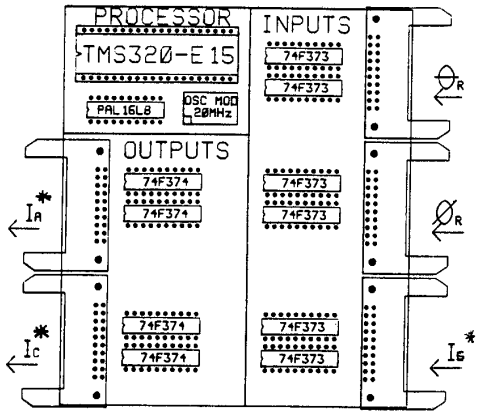

Figure 3. PCB Layout for vector controller

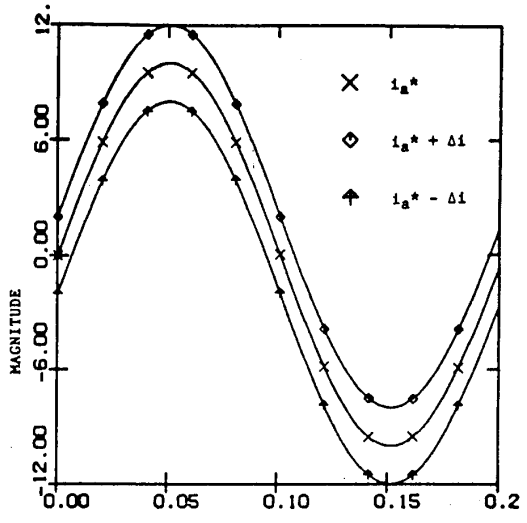

Figure 4. Hysteresis current controller

The logic for the implementation of an hysteresis current controller is shown in Table 1 for phase A, where the superscript * denotes the commanded current.

Table 1

$i_{a*} \geq 0$	$i_a < i_{a*}^* - h$	T1 on
$i_{a*} \geq 0$	$i_a > i_{a*}^* + h$	T1 off
$i_{a*} < 0$	$i_a > i_{a*}^* + h$	T4 on
$i_a^* < 0$	$i_a < i_a^* - h$	T4 off

where h is the hysteresis size $\equiv \Delta i$.

This was programmed using TI DSP assembly language in scaled integer arithmetic. In order to reduce the requirements for sensors, only phase A and C currents were measured. In order to reduce the processing time of the vector controller loop, only the commanded phase A and C currents are calculated. Hence both the commanded and measured phase B currents are calculated first in the current controller from the fact that $i_a + i_b + i_c = 0$. The current controller algorithm was implemented sequentially for phases A,B and C.

The DSPs work in two's complement arithmetic. Although A/D converters exist which give two's complement output, the high speed, 3 microsecond A/D converters used gave an unsigned binary output. This had to be converted to two's complement for correct manipulation by the DSP. The assembly language code for the 320E15 is quite restricted and the logic in Table 1 had to be rearranged so that the instructions available could be used.

The general purpose PCB which was designed for the vector controller was modified slightly for the current controller as shown in figure 5. The input data latches used in ports 0 and 1 of the vector controller were replaced with the A/D converters which had a tri-state output. Only when the A/D converter was enabled from the PAL decoder did its output appear on the data bus for a long enough time for the DSP to read the output. Three separate ports were used to control the firing of the transistors in each phase. A future implementation will look at the possibility of using just one port for this purpose at the expence of increased processing time.

6. PROGRAMMING THE DSPS

The TI EVM development system was used to program the DSPs. Although it was designed to program the C10 devices, it is quite capable of programming the C15/C17 devices, using adapters that come with the EVM.

The assembly language DSP program can be assembled line by line as it is written and stored directly into the EVM memory where it can be debugged with breakpoints or single stepping of the program. Although the program can then be stored on audio tape, it is also possible to communicate between the EVM and a host computer through a RS232 interface. This allows the DSP program to be written using an editor on the host system which can then be downloaded to the EVM. The EVM has the capability of assembling the incoming assembly language program, line by line and storing the result in memory. In this design, the host system was a Codata computer running Unix. The eprom/rom programmer is on board the EVM system.

Perhaps the most useful feature of the EVM is the capability for emulation. It is possible to connect the EVM into the designed PCB in place of the DSP and single-step through the program or run the program with breakpoints. Hence the DSP program can be entirely tested before finally programming the DSPs.

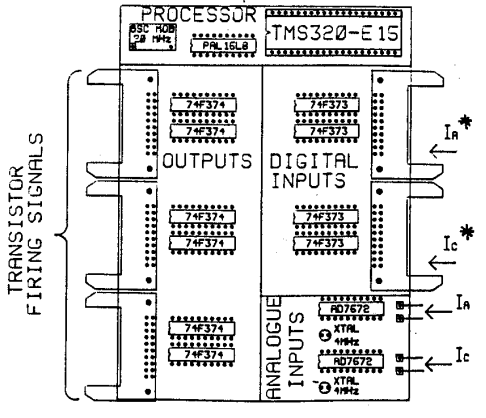

Figure 5. PCB Layout for current controller

The choice of the particular device depends on the specific application, including the number of input/output ports, the need for serial interfacing and processing speed. In this paper, all four devices mentioned above are evaluated for the vector controller.

4. DSP-BASED VECTOR CONTROL

The implementation of the vector controller requires that the instantaneous rotor angle θ_r from the resolver, the amount of phase advance angle ϕ and the magnitude of the commanded stator current vector i_s be read into RAM. The two angles are then added with the result left in the accumulator. The cosine of this angle is then calculated by using the TBLR instruction which uses the angle in the accumulator as the program memory address to look up the cosine. This is then multiplied by i_s to yield the commanded phase A current. The commanded current of phase C is calculated in a similar way except 120^0 is added to $\theta_r+\phi$ before looking up the cosine. Although the commanded phase C current can be obtained by adding 240^0 to $\theta_r+\phi$, it is easier to use the fact that the sum of the three currents must be zero for a balanced, three phase system.

When the DSP is reset, the program counter points to address zero. Hence either the DSP program must begin here or an unconditional branch to the program address be provided; for reasons given later, the former was decided. For the specification of 5.3 arc-minutes of resolution and a 2-pole resolver, a 12-bit resolution in the R/D converter is required. Hence 2^{12}=4096 values of cosine must be available in the DSP. Since there is only 4k of memory altogether, all cosine values cannot be stored. Fortunately, the cosine function is periodic and only 90^0=1024 values need to be stored. Tests can then be conducted on the angle to determine which quadrant it is in and hence the sign of the cosine. This requires a longer program to deal with the cosine values outside the range, resulting in increased execution time.

It was found that the vector control algorithm took up less than 1k of memory, hence it was possible to store 270^0 of look-up table and hence reduce the amount of checking to be done. The total number of cycles needed for execution is 50. The specification of 3000 rpm = 50 Hz, results in the bit rate from the encoder to be 50 x 4096 = 204800 pulses/sec, ie a period of $4.9\text{x}10^{-6}$s. It is immediately clear that the TMS320E15-25/C15-25 which operate at 160ns can only support $4.9\text{x}10^{-6}/160\text{ns}$ = 30 machine cycles of code before which the program should branch back to its starting point. The program in appendix 1 takes 50 implying that the TMS320E15/C15 is unsuitable for this implementation. Similarly, the TMS320E17, which has a 200 ns cycle time can only support 24 machine cycles of code which is also too few for the above algorithm. The ROM-based 100ns C17 device is just outside the specification, since it can support 49 machine cycles of code. Hence the next design presented is an adaptation of the previous to speed up the program cycle time.

The approach taken was to determine the relationship between the position encoder resolution and the program cycle time. The required resolution in the position is also achievable with a reduced resolution in the R/D converter but to use a multipole resolver. An 8-pole resolver with a 10-bit resolution in the digital word would also meet the position specification. The reason is that for a given mechanical rotation, the 8-pole resolver experiences 4 electrical rotations which when multiplied by 2^{10} = 4096 counts in 360 mechanical degrees. For a given mechanical speed, the bit rate is the same as for the 12 bit R/D converter.

Since now only a 10 bit representation of the angle is used, the full 360^0 can be stored in 1024 words of memory. The range of the angle from the R/D converter is 0 to 1023. The angle ϕ which ranges from 90^0 to 180^0 (512 in scaled integer arithmetic) is then added to the R/D output. This angle is then used to calculate the commanded phase A current. In addition, 120^0 (341 in scaled integer arithmetic) is also added to the above to calculate the commanded phase C current. Hence the maximum possible angle is 360+180+120 = 660^0 ie 1023+512+341 = 1876 in scaled integer arithmetic. But this amount of eprom space is available, hence in this second design, all 660^0 of lookup table is stored. This results in no requirement for the checking of the quadrant of the angle with a substantial saving in program cycle time.

The no. of machine cycles needed is 35 which for the 160ns cycle time of the C15/E15 devices, is a program cycle time of $5.6\text{x}10^{-6}$s which is slightly larger than the desired $4.9\text{x}10^{-6}$s. This would therefore support a machine speed of 2625 rpm. Hence a second modification of this design is done so as to meet the specification as discussed next.

The generation of the third phase current (which is obtained by adding the reference currents described above and negating the result) consumes 7 machine cycles. The modification made is to transfer this processing load to the current controller and to output just the commanded phase A and C currents. This results in a program cycle time of $4.48\text{x}10^{-6}$s, with a resulting machine speed of 3281 rpm which is now inside the specification.

It was explained previously that the position information is an absolute value between 0 and 1023. The program was stored from the zero memory location and the cosine table from 1024. This relative shift could have been accomplished in the program by adding 1024 to the angle at the expence of an increase in the execution time. A hardware alternative is to connect the 11th bit on the input port reading in the rotor angle, permanently high. This automatically shifts the zero angle from 0 to 1024 and avoids a software addition.

The input/output ports of the DSP are connected to the rest of the drive system through latches. A PAL decoder is used to decide on the particular latches to be enabled. A general purpose pcb was designed and built with the system configured to have 4 input and 4 output ports in the case of the E15/C15 devices as shown in figure 3. The reasoning was to use the same pcb and associated hardware for both the vector and current controllers, with just a change in the DSP program.

5. DSP-BASED CURRENT CONTROLLER

Current control algorithms have always been difficult to implement digitally, because of the high bandwidth and hence sampling required. Hysteresis and ramp comparison current controllers have been used for vector controlled drives. In this paper, an hysteresis current controller is implemented as shown in figure 4.

7. RESULTS

The programmed DSPs were tested both on an R-L load and a 1.5kW permanent magnet synchronous motor drive. To test the DSPs on an R-L load, additional test circuitry was built which included a counter to simulate the resolver to digital converter output. In addition, a DAC was used to check the validity of the vector controller output.

The vector controller output was fed into the current controller. A single- phase test inverter circuit was used to drive the R-L load from one of the current controller output ports. The actual current flowing in the R-L load was measured, fed through the A/D converter on the current controller board and used by the algorithm to force the actual current to follow the commanded.

The programmed DSPs were tested both on an R-L load and a 1.5kW permanent magnet synchronous motor drive. To test the DSPs on an R-L load, additional test circuitry was built which included a counter to simulate the resolver to digital converter output. In addition, a DAC was used to check the validity of the vector controller output.

The vector controller output was fed into the current controller. A single- phase test inverter circuit was used to drive the R-L load from one of the current controller output ports. The actual current flowing in the R-L load was measured, fed through the A/D converter on the current controller board and used by the algorithm to force the actual current to follow the commanded.

Figure 6 shows the commanded and actual currents in the R-L load at a commanded frequency of 200Hz. This corresponds to a mechanical speed of 3000 rpm for an 8- pole machine, 6000 rpm for a 4-pole machine or 12000 rpm for a 2-pole machine. The upper and lower transistor base drive signals are also shown. The pulse-width-modulation of the transistors are evident as the controller attempts to control the actual current. Figure 7 shows the results during saturation of the current regulator, when the system runs out of dc bus voltage to force the actual current to follow the commanded. During saturation, the transistors are turned fully on to make the best use of the available bus voltage.

Figure 8 shows the results when the DSP based vector and current controllers drive a 1.5kW machine. Here the resolver to digital converter output is fed into the DSP doing the vector control. The upper trace is one output of the vector controller, namely the commanded phase A current. This is fed into the current controller, which outputs the upper and lower transistor base drive signals of phase A of the three phase inverter. These are the lower two traces in figure 8. The trace below the commanded current is the actual motor current, controlled by the DSP-based current controller. The electrical period is 15ms which corresponds to an electrical frequency of 66.67 Hz. This therefore implies a mechanical speed of 4000 rpm for a 2-pole machine, 2000 rpm for a 4-pole machine and 1000 rpm for an 8-pole machine.

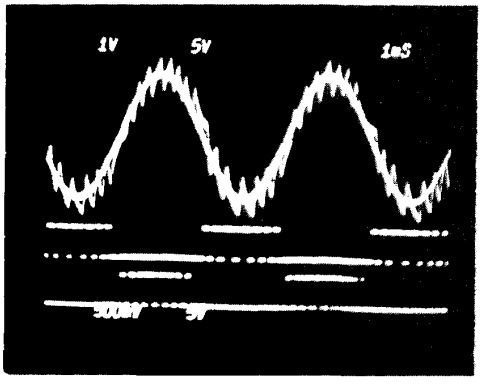

Figure 6. R-L load results

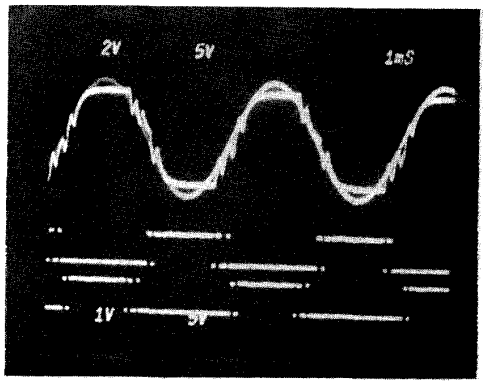

Figure 7. R-L load results during saturation of the current regulator

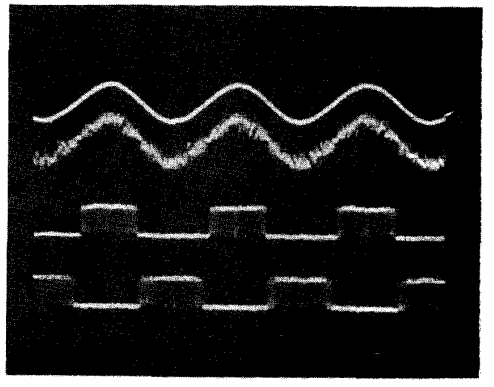

Figure 8. PMSM results for an electrical period of 15ms

Figure 9 shows the results for an electrical period of 9 ms which corresponds to 111 Hz or 6667 rpm for a 2-pole machine, 3333 rpm for a 4-pole and 1667 rpm for an 8-pole machine. Current control at this higher frequency is slightly worse than at the lower electrical frequency because of the lower number of switches per cycle in the former.

8. CONCLUSIONS

This paper has made a detailed evaluation of the use of DSPs for vector and current control in PMSM drives. Although faster DSPs with larger instruction sets are available, the TMS320E15 devices were used because of the on-board memory and ease of development. In the case of the vector controller, the following conclusions can be drawn.

The E15/C15 and E17/C17 devices were not able to meet a standard industrial specification of 3000 rpm at 5.3 arc minutes of resolution if a 2 pole resolver and 12 bit R/D converter is used. The 100 ns ROM-based C17 was just outside the specification.

The E15/C15 and C17 devices can meet the specification if a 8 pole resolver and a 10 bit R/D converter is used. This is accomplished by storing all possible angles needed in memory to avoid the overhead of checking for the quadrant that the angle is in.

The above indicates that the position feedback should be chosen carefully in relation to the vector controller. This calls for an integrated approach to drive design and not for each component to be designed or selected in isolation.

In this design, it was found advantageous to share the processing load between two DSPs. The first one does the vector control with the second responsible for the current control. A higher speed current controller would be possible if a DSP were used for each phase or a faster DSP used with external memory. The latter option would be at the expence of an increased pin count and hence lower reliability. Practical results on an R-L load and an 8-pole PMSM demonstrate the ability of DSPs to be used for both the vector and current controllers in PMSM drives. At a given mechanical speed, the lower the pole number, the lower is the commanded electrical frequency and the better the current control from the DSP.

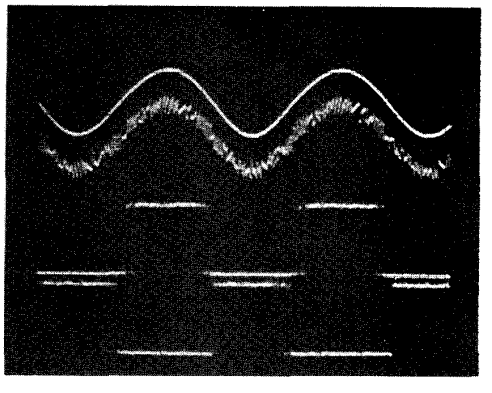

Figure 9. PMSM results for an electrical period of 9ms

Acknowledgements

P.Pillay would like to acknowledge the SERC for funding his research. The authors acknowledge P.Freere for his help in testing the DSP designs.

REFERENCES

[1] M.F. Rahman, T.S. Low and L.B. Wee, "Development of a digitally controlled permanent magnet brushless dc drive system," Conference on Applied Motion Control, 1986, pp 283-288.

[2] B.K.Bose and P.M.Szczesny, "Microcomputer-based control and simulation of an advanced interior permanent magnet (IPM) synchronous machine drive system for electric vehicle propulsion", IEEE Trans., vol IE 35, No.4, pp 547-559.

[3] B.K. Bose, "A high performance inverter-fed drive system of an interior permanent magnet machine", 1987 IEEE IAS Annual Meeting, pp 269-276.

[4] N.Matsui and H.Ohashi, "DSP-based adaptive control of a brushless motor", 1988 IAS Annual Meeting, pp 375-380.

[5] P.Pillay and R.Krishnan, "Modeling, analysis and simulation of permanent magnet motor drives-Part 1: The permanent magnet synchronous motor drive" IEEE Transactions on Industry Applications, March/April 1989, pp 265-273.

[6] G.Pfaff, A.Weschta and A.Wick, "Design and experimental results of a brushless ac servo drive" IEEE IAS Annual Meeting, 1982, pp 692-697.

[7] T.M.Jahns, G.B. Kliman and T.W.Neumann, "Interior permanent magnet synchronous motors for adjustable speed drives", IEEE IAS Annual Meeting, 1985, pp 814-823.

[8] T.Sebastian and G.R.Slemon, "Operating limits of inverter-driven permanent magnet motor drives", IAS Annual Meeting, 1986, pp 800-803.