High Speed Digital Distance Protection - Real Time Simulation and Hardware Development

K. K. Li+ C. Cheung+ Y. Q. Xia++

+ Hong Kong Polytechnic, Kowloon, HONG KONG
++Tianjin University, Tianjin, P. R. CHINA

Abstract: This paper presents the development of a high speed digital distance protection scheme based on the finite Fourier transform algorithm. Techniques on cutting down on-line calculation are introduced. Real-time simulation shows that all the relay processing function can be completed within one sample interval based on 4kHz sampling rate. A multi-processor hardware scheme is developed. Testing result shows that the protective relay possesses very fast operation with high sensitivity and selectivity.

1. Introduction

Protection of equipments and transmission lines in extra-high-voltage power system generally requires advanced relaying schemes. These relaying schemes usually need high processing capability to deal with today's power system requirements. High speed fault clearance also appears more and more important under fault conditions[1]. For these reasons there has been a growing effort on high speed digital transmission line protection[2]-[6].

To explore the feasibility of developing a high speed digital protective relay using general purpose microprocessors, previous works have been done at relaying algorithm based on modified finite transform[7][8]. According to the simulation of the relaying algorithm, high data sampling rate is needed to manifest the advantages of the algorithm in fast convergency. Simulation results show that the relay can operate correctly within 7 ms even in the worst situation if a 4kHz sampling rate is used. To exploit the most update sampled data and make the relay respond immediately to the processing result within one sampling interval, all the digital filtering, impedance calculation and logic judgment must be completed within 0.25ms. For this purpose, the authors concentrate their effort in two aspects: first is on real-time simulation in which the main aim is to reduce the on-line calculation, and secondly is on hardware development and the intelligent implementation of the relay function on a 80286 computer board, together with a data acquisition board working in parallel.

2. Relaying Algorithm [7,8]

The transmission line being protected is modeled by a set of first order linear differential equations in the form,

$$v_r(t) = R\, i_r(t) + L\, di_r(t) / dt \qquad (1)$$

where

$v_r(t)$, $i_r(t)$ are instantaneous value of relaying voltage and current.

R, L are measured resistance and inductance.

Application of finite Fourier transform to eqn.1 produces the relationship,

$$\bar{v}_r(j\omega_e,t) = R\, \bar{i}_r(j\omega_e,t) + j\omega_e L\, \bar{i}_r(j\omega_e,t) \\ +L\, \{i_r(t)\exp(-j\omega_e t) - i_r(t-T_w)\exp[-j\omega_e(t-T_w)]\} \qquad (2)$$

where

$\bar{v}_r(j\omega_e\ ,\ t)$, $\bar{i}_r(j\omega_e\ ,\ t)$ are finite Fourier transform of $v_r(t)$ and $i_r(t)$.

T_w is finite Fourier transform window duration.

Using their decomposed real and imaginary parts instead of $\bar{v}_r(j\omega_e,t)$, $\bar{i}_r(j\omega_e\ ,\ t)$ in eqn.2, the measurement of the line impedance is achieved in a matrix form,

$$\begin{bmatrix} R \\ L \end{bmatrix} = \frac{1}{D}\begin{bmatrix} S_i & -C_o \\ -\bar{i}_{r2} & \bar{i}_{r1} \end{bmatrix}\begin{bmatrix} \bar{v}_{r1} \\ \bar{v}_{r2} \end{bmatrix} \qquad (3)$$

where

$$\bar{v}_{r1} = \int_{t-T_w}^{t} v_r(\tau)\cos(\omega_e\tau)\, d\tau \qquad (4)$$

$$\bar{v}_{r2} = \int_{t-T_w}^{t} v_r(\tau)\sin(\omega_e\tau)\, d\tau \qquad (5)$$

$$\bar{i}_{r1} = \int_{t-T_w}^{t} i_r(\tau)\cos(\omega_e\tau)\, d\tau \qquad (6)$$

$$\bar{i}_{r2} = \int_{t-T_w}^{t} i_r(\tau)\sin(\omega_e\tau)\, d\tau \qquad (7)$$

$$C_o = i_r(t)\cos(\omega_e t) - i_r(t-T_w)\cos[\omega_e(t-T_w)] + \omega_e\bar{i}_{r2} \qquad (8)$$

$$S_i = i_r(t)\sin(\omega_e t) - i_r(t-T_w)\sin[\omega_e(t-T_w)] - \omega_e\bar{i}_{r1} \qquad (9)$$

$$D = \bar{i}_{r1}S_i - \bar{i}_{r2}C_o \qquad (10)$$

3. Real Time Simulation

The results of the relaying algorithm simulation in high level language shows that the algorithm possesses rapid convergency and stable characteristics with a very short data window. The relay using this algorithm can operate very quickly under any fault condition if 4kHz sampling rate is exployed and all the calculations can be completed within one sampling interval (0.25ms). For this purpose, cutting down on-line calculation is the main task.

3.1 Discrete digital processing

The measured resistance and reactance applicable to any sampling instant k are determined from eqn.3 which, when written in discrete form, yields R(k) and X(k) as follows:

$$R(k) - [S_i(k)\ \bar{v}_{r1}(k)\ -\ C_o(k)\ \bar{v}_{r2}(k)]\ /\ D(k) \quad (11)$$

$$X(k) - 2\pi f L(k) - 2\pi f[\bar{i}_{r1}(k)\bar{v}_{r2}(k) - \bar{i}_{r2}(k)\bar{v}_{r1}(k)]/D(k) \quad (12)$$

The three phase voltages and currents are sampled and converted to digital form. The corresponding values of $v_r(t_k)$, $i_r(t_k)$ are combined according to different fault type, where t_k represents the instant at any kth sample of the signal. To get the value of R(k) and X(k), the first step is to calculate $\bar{v}_{r1}(k)$, $\bar{v}_{r2}(k)$, $\bar{i}_{r1}(k)$, $\bar{i}_{r2}(k)$, $C_o(k)$, $S_i(k)$, and D(k). With reference to eqn.4 to eqn.10, they can be derived through discrete digital processing in consideration of saving on-line computation time.

$$\begin{aligned}\bar{v}_{r1}(k) = \bar{v}_{r1}(k-1) + [v_r(t_k) + v_r(t_{k-1})]\,C_1(k)\\ -[v_r(t_{k-N}) + v_r(t_{k-1-N})]\,C_1(k-N)\end{aligned} \quad (13)$$

$$\begin{aligned}\bar{v}_{r2}(k) = \bar{v}_{r2}(k-1) + [v_r(t_k) + v_r(t_{k-1})]\,S_1(k)\\ -[v_r(t_{k-N}) + v_r(t_{k-1-N})]\,S_1(k-N)\end{aligned} \quad (14)$$

$$\begin{aligned}\bar{i}_{r1}(k) = \bar{i}_{r1}(k-1) + [i_r(t_k) + i_r(t_{k-1})]\,C_2(k)\\ -[i_r(t_{k-N}) + i_r(t_{k-1-N})]\,C_2(k-N)\end{aligned} \quad (15)$$

$$\begin{aligned}\bar{i}_{r2}(k) = \bar{i}_{r2}(k-1) + [i_r(t_k) + i_r(t_{k-1})]\,S_2(k)\\ -[i_r(t_{k-N}) + i_r(t_{k-1-N})]\,S_2(k-N)\end{aligned} \quad (16)$$

$$C_o(k) - i_r(t_k)\,C_2(k) - i_r(t_{k-N})\,C_2(k-N) + \omega_e \bar{i}_{r2}(k) \quad (17)$$

$$S_i(k) - i_r(t_k)\,S_2(k) - i_r(t_{k-N})\,S_2(k-N) - \omega_e \bar{i}_{r1}(k) \quad (18)$$

$$D(k) - \bar{i}_{r1}(k)\,S_i(k) - \bar{i}_{r2}\,C_o(k) \quad (19)$$

where

$$\begin{aligned}C_1(k) &= \sin(\omega_e t_k) - \sin(\omega_e t_{k-1}).\\ S_1(k) &= \cos(\omega_e t_k) - \cos(\omega_e t_{k-1}).\\ C_2(k) &= \cos(\omega_e t_k)\\ S_2(k) &= \sin(\omega_e t_k)\end{aligned}$$

A quadrilateral type characteristic and counting technique [3],[7],[8] are used in this digital relay scheme. The reach boundary constraining equations are presented in the appendix. The usual method of determining the relay operation is to allow the processor successively calculate the measured impedance at each sample instant and then compare them with boundary constraints. This method is simple but it requires more on-line calculation time. Therefore, faster processing techniques are required.

3.2 Fast processing techniques

To complete all the calculations and boundary comparison in one sampling interval is a hard task for a common 16-bit microprocessor. The target microprocessor (80286) can provide instruction sets to perform 16-bit-by-16-bit integer multiplication and 32-bit value by 16-bit value division. There is no instruction set support on long word multiplication and division. Using floating point processing routine to perform the relaying computation would vastly exceed the time limit. Utilization of math coprocessor can cut down long word multiplication and division executing time, but a testing program running under real time simulation mode indicates that all the computation cannot be completed within the time limit even with a math coprocessor. To reach a better solution, a practical less on-line calculation approach is presented here.

In eqn.13 to eqn.18, $v_r(t_k)$, $i_r(t_k)$ are obtained from 12-bit A/D conversion. $C_1(k)$, $C_2(k)$, $S_1(k)$, $S_2(k)$ are calculated off-line and scaled to proper length integers (in a word length) with enlarging coefficients. The enlarging coefficients are calculated and selected according to preciseness requirement and convenience of restoring their original values by using simple right shift instruction. Since the value of $C_1(k)$, $C_2(k)$, $S_1(k)$, $S_2(k)$ are always smaller than 1, the results of $\bar{v}_{r1}(k)$, $\bar{v}_{r2}(k)$, $\bar{i}_{r1}(k)$, $\bar{i}_{r2}(k)$, $S_i(k)$ and $C_o(k)$ can be easily derived and scaled to one-word length integer by using 8086 fast assembly instructions. From eqn.19, D(k) will be in double word length. Because of absence of 32-bit division instruction in assembly instruction set, direct calculation of R(k) and X(k) (eqn.11, 12) and the comparison of the relay setting (eqn. 34, 35 in Appendix) will take a long time. A change of form is therefore required.

Take eqn.12 and one constraint term in eqn.34 of Appendix (X_o < X(k) < $0.8X_r$) as an example. They can be combined to give the form below:

$$X_o < 2\pi f[\bar{i}_{r1}(k)\bar{v}_{r2}(k) - \bar{i}_{r2}(k)\bar{v}_{r1}(k)]/D(k) < 0.8X_r \quad (20)$$

It can be rewritten as:

$$\frac{X_o}{2\pi f}D(k) < \bar{i}_{r1}(k)\bar{v}_{r2}(k) - \bar{i}_{r2}(k)\bar{v}_{r1}(k) < \frac{0.8X_r}{2\pi f}D(k), \quad D(k)>0 \quad (21)$$

$$\frac{X_o}{2\pi f}D(k) > \bar{i}_{r1}(k)\bar{v}_{r2}(k) - \bar{i}_{r2}(k)\bar{v}_{r1}(k) > \frac{0.8X_r}{2\pi f}D(k), \quad D(k)<0 \quad (22)$$

Similarly, the other constraint terms may change their form, and eqn.34 and eqn.35 becomes:

(i). when D(k) < 0

$$\begin{cases} a_1x(k) > D(k) > a_2x(k)\\ a_3D(k) > r(k) > a_4D(k) + a_5x(k)\\ D(k) < a_6x_m(k)\end{cases} \quad (23)$$

$$\begin{cases} a_2x(k) > D(k) > a_7x(k) \\ a_3D(k) > r(k) > a_4D(k) + a_5x(k) \\ D(k) < a_6x_m(k) \end{cases} \tag{24}$$

(ii). When D(k) > 0

$$\begin{cases} a_1x(k) < D(k) < a_2x(k) \\ a_3D(k) < r(k) < a_4D(k) + a_5x(k) \\ D(k) > a_6x_m(k) \end{cases} \tag{25}$$

$$\begin{cases} a_2x(k) < D(k) < a_7x(k) \\ a_3D(k) < r(k) < a_4D(k) + a_5x(k) \\ D(k) > a_6x_m(k) \end{cases} \tag{26}$$

where
a_1 to a_7 and r(k), x(k), x_m(k) are expressed in eqn.27.

$$\begin{cases} r(k) = S_i(k)\bar{v}_{r1}(k) - C_o(k)\bar{v}_{r2}(k) \\ x(k) = \bar{i}_{r1}(k)\bar{v}_{r2}(k) - \bar{i}_{r2}(k)\bar{v}_{r1}(k) \\ x_m(k) = \bar{i}_{r1}(k)\bar{v}_{r2}(k-N_T) - \bar{i}_{r2}(k)\bar{v}_{r1}(k-N_T) \\ a_1 = \dfrac{2\pi f}{0.8X_r}, \quad a_2 = a_6 = \dfrac{2\pi f}{-X_o}, \quad a_3 = -R_o \\ a_4 = R_r, \quad a_5 = 2\pi f K_1, \quad a_7 = \dfrac{2\pi f}{X_r} \end{cases} \tag{27}$$

The factors (a_1 to a_7) are pre-calculated off-line. The on-line multiplication of long word becomes combination of some shift and addition. Consider a_1x(k) as an example, it can be derived from 8086 fast assembly instructions expressed as follow.

$$\begin{aligned} a_1x(k) = &\sum_{i=0}^{M_1} \alpha_{l1i}[i \textit{ bit leftshift of } x(k) \\ &+\sum_{i=1}^{m_1} \alpha_{r1i}[i \textit{ bit rightshift of } x(k)] \end{aligned} \tag{28}$$

where

M_1 is maximum numbers of left shift bit,
m_1 is maximum numbers of right shift bit,
α_{l1i}(i=0,1,...,M_1), α_{r1i}(i=1,2,...,m_1) are true-false factors (1 or 0).

These factors can be calculated off-line according to setting values. M_1 is expressed as:

$$M_1 = Ceil(\log_2 a_1 - 1) \tag{29}$$

Ceil($\log_2$ a_1 -1) represents the smallest integer not less than ($\log_2$ a_1 -1).

m_1 is determined according to the value of M_1 and the requirement of calculating preciseness.

$$m_1 = C_m - M_1 \tag{30}$$

where C_m is a preciseness-determining constant. It is selected and determined according to requirement of calculating preciseness and the amount of on-line computation.

The maximum relative possible error E_m caused by the digital arithmetic is:

$$E_m = \frac{0.5}{2^{C_m} - 1} \tag{31}$$

Let C_m=8, the maximum relative error is under 0.2%. The true-false factors α_{l1i} and α_{r1i} are expressed:

$$\alpha_{l1i} = \begin{cases} 1, & (a_1 - \sum_{j=0}^{M_1-i-1} 2^{M_1-j}) > 2^i \\ 0, & (a_1 - \sum_{j=0}^{M_1-i-1} 2^{M_1-j}) < 2^i \end{cases} \quad (i = 0,1,2,...M_1-1) \tag{32}$$

$$\alpha_{r1i} = \begin{cases} 1, & (a_1 - \sum_{j=0}^{M_1-i-1} 2^{M_1-j}) > 2^i \\ 0, & (a_1 - \sum_{j=0}^{M_1-i-1} 2^{M_1-j}) < 2^i \end{cases} \quad (i = -1,-2,...,m) \tag{33}$$

M, m, α_{li}, α_{ri} of the other factors (a_2 to a_7) can be calculated off-line in the same way. For a typical 128km 400kV double circuit overline, its setting values are X_r = 29.4 Ω, X_o = -4.0 Ω, R_r = 29.4 Ω, R_o = -4.0 Ω, $\tan^{-1}$ (X_{1L}/R_{1L}) = 84° [7], [8], the corresponding factors are calculated off-line and listed in table I.

Table I Setting factors derived off-line (C_m = 8)

	a_1	a_2	a_3	a_4	a_5	a_6	a_7
value	13.36	78.5	4.0	29.4	33.01	78.5	10.69
M	3	6	2	4	5	6	3
m	5	2	6	4	3	2	5
α_{l6}	0	1	0	0	0	1	0
α_{l5}	0	0	0	0	1	0	0
α_{l4}	0	0	0	1	0	0	0
α_{l3}	1	1	0	1	0	1	1
α_{l2}	1	1	1	1	0	1	0
α_{l1}	0	1	0	0	0	1	1
α_{l0}	1	0	0	1	1	0	0
α_{r1}	0	1	0	0	0	1	1
α_{r2}	1	0	0	1	0	0	0
α_{r3}	1	0	0	1	0	0	1
α_{r4}	0	0	0	0	0	0	1
α_{r5}	0	0	0	0	0	0	0
α_{r6}	0	0	0	0	0	0	0

Table I lists the values of setting factors under a general situation. To further reduce the on-line computation time, a_2, a_3, a_5, a_6 are modified and fixed in constant values based on relay setting. For example, if a_5 equals 32, the arithmetic of a_5x(k) can be quickly completed by left shifting x(k) five bits. In this scheme, let a_2 = a_6 = 128, a_3 = 4, a_5 =32. The values of X_0, R_0, $\tan^{-1}$ (X_{L1}/R_{L1}) will be modified correspondingly to -2.45Ω, -4Ω, 84.2°.

3.3 Real-time simulation result

Real-time simulation for various faults has been completed on a PC AT computer. The input data of fault transient voltages and currents are generated by a Transient Waveform Simulation Program (TWSP) developed based on [9]. The generated fault data is loaded into a pre-processing routine which converts the floating point data into straight binary data format. The constant coefficients $C_1(k)$, $C_2(k)$, $S_1(k)$, $S_2(k)$ in one cycle are precalculated and stored in memory. The factors in Table I are derived from a setting routine. The relay real-time function routine is written in assemble language with possible fast instructions. On-line processing time is counted precisely by accessing 8254 timer chip on board. Results show that all the required on-line processing can be completed within 250 microseconds based on a 12MHz 80286 CPU. A typical computation time is 150 microseconds.

4. Hardware Implementation

The hardware is implemented for testing by utilising a PC AT compatible computer board. A 12-bit A/D card controlled by a micro-controller (8796BH) is fabricated to complete analog to digital conversion in parallel. The overall block diagram of the digital relay is shown in fig.1.

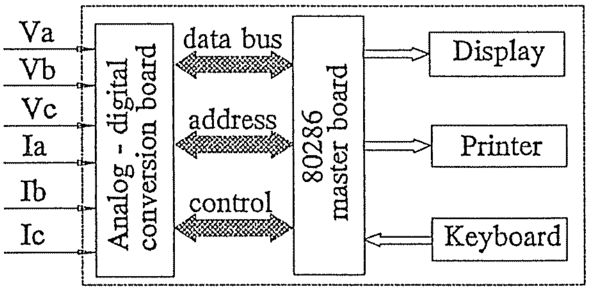

Figure 1 The overall block diagram of the relay

The hardware was carefully designed, in order to achieve the following objectives:

1). The 80286 computer will concentrate on the numerical intensive task of relay function calculation based on the finite Fourier transform algorithm.

2). All the data acquisition and preprocessing tasks are completed by the 8796BH microcontroller on the A/D board.

3) Data transfer between the A/D board and the 80286 computer should be simple, reliable, and consume minimal amount of processing time.

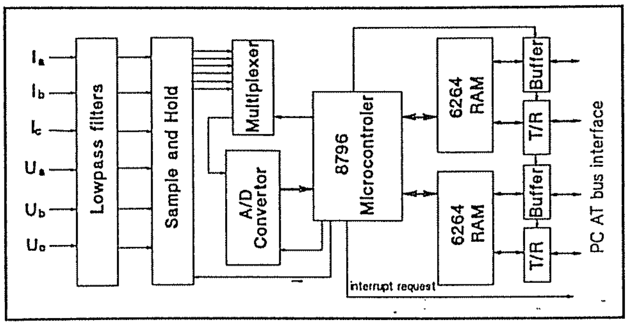

Figure 2 A/D hardware block diagram

A block diagram of A/D board is shown in Fig.2. Its major functions are scaling, anti-aliasing filtering, sample-and-hold, multiplexing, fast analog to digital conversion in 12-bits and data pre-processing. Through second-order lowpass Butterworth filters (2kHz cutoff frequency), the three scaled voltage and three current signals are sampled and hold simultaneously. Six S/Hs (LF398) are used to hold the analog input signals constant during A/D conversion to avoid any errors due to rapid fluctuation in the signals. A multiplexer is used to select the six signals for A/D conversion in turn. The A/D convertor chip (AD ADC84) is a high speed low cost successive approximation convertor which includes an internal clock reference and comparator. Its conversion time is $10\mu s$. A 16-bit micro-controller, INTEL 8796BH, is used to coordinate all the activities on the board. Operating directly on a 256-byte register space, the CPU is able to quickly change context at a clock frequency of 12MHz. 8k byte internal on-chip program memory is available. Sufficient utilization of the provided functions from the microcontroller chip greatly simplifies the hardware structure of the A/D board. Under the control of the 8796 micro-controller, each converted digital data is transferred to the internal registers. After data pre-processing in the registers, six scaled voltage and current data of one word length are transferred to the two RAM (6264) chips. The RAMs can be commonly accessed (read or write) by either 8796BH or 80286 processor through two groups of tri-state data buffer and bus latching circuits. It occupies a physical address of C8000 -- CFFFF in the 80286 computer board. One bit of a I/O port on the microcontroller is used to control the memory access selection. After the completion of all six channels' A/D conversion and data preprocessing, the microcontroller will release an interrupt single to inform the 80286 processor to transfer data from the common memory into its own registers for relay function computation. All calculations should be completed before the next interrupt cycle.

5. Software strategy

The software is divided into two parts. The first part is written for the 8796 microcontroller. It contains routines for all the hardware functional block control and data acquisition. The second part which contains the relay function calculation and display module, is written in

8086 format.

Software package for the A/D board, written in 8096 assembly language (ASM-96), is resident in the internal on-chip EPROM of the 8796 microcontroller. It consists of two routines, initialization routine and interrupt service routine. Registers, flags, I/O port state and timers are initialized in the initialization routine. The sampling interval is set by the internal hardware timer. For every inter-sampling period (250μs), the interrupt routine is triggered to process the A/D conversion and data preprocessing. At the beginning of the interrupt service routine, the shared RAMs will switch to 8796BH for access, but it will switch back to the 80286 computer toward the end of the interrupt routine.

The 8086 software package is a mixed-language programming package in which the off-line calculating and displaying modules are written in high level language C with some inline assembly codes. The relay on-line processing functions is written in 8086 assembly language (ASM-86) in consideration of faster execution time.

The off-line calculating and displaying C modules consist of the following routine:

i) Relay setting routine,
ii) constant coefficients (C1, C2, S1, S2) calculating and scaling routine,
iii) setting factors calculating routine,
iv) data processing routine, and
v) waveform and testing result display routine.

Real-time relay function in ASM-86 module consists of:
i) Waveform recording routine,
ii) $\overline{v}_{r1}(k)$, $\overline{v}_{r2}(k)$, $\overline{i}_{r1}(k)$, $\overline{i}_{r2}(k)$, $C_o(k)$, $S_i(k)$, and $D(k)$ calculating routine,
iii) $r(k)$, $x(k)$, $x_m(k)$ calculating routine,
iv) relay boundary restraint performing routine, and
v) relay's operation routine.

6. Real-time testing

Real-time testing was completed in laboratory by utilizing a transient waveform generator showed in fig.3. The computer-controlled waveform generating unit consists of a digital to analog board and an amplification subunit. The prefault and postfault data are obtained from the Transient Waveform Simulation Program (TWSP). Through D/A conversion and proper amplification, three phase voltage and current analog signals used as the input signals of the relay under testing are generated. Various fault waveforms are generated according to different fault types.

The rely was tested for different fault conditions. Some testing results of phase-A to ground faults are shown in fig.4 to fig.7. The waveforms and the relay output are recorded on-line through the waveform recording routine and are displayed on a colour monitor screen through the waveform displaying routine. In the figures, the solid lines are three phase voltages (v_a, v_b, v_c) and the dash lines are three phase currents(i_a, i_b, i_c). The values on the Y axis are in p.u. of v and 0.2i. The mark "▾" indicates the fault inception instant. The transient waveforms are generated based on a 400kV line of length 128km interconnecting two power sources [3,7,8], The first zone reach of the relay is set to 80% of the line length -- 102km.

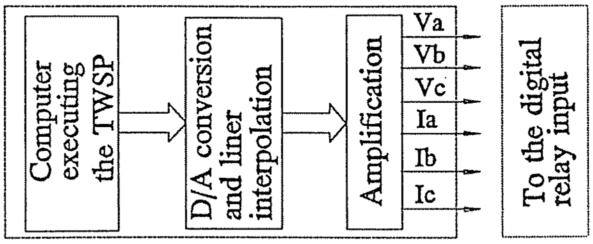
Computer executing the TWSP
D/A conversion and liner interpolation
Amplification
Va
Vb
Vc
Ia
Ib
Ic
To the digital relay input

Figure 3 Waveform generator block diagram for testing

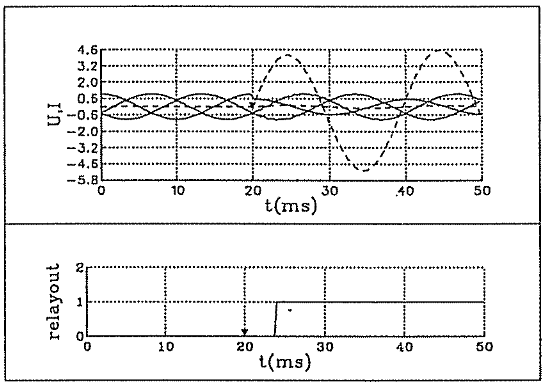

Figure 4 Fault at 12.8km

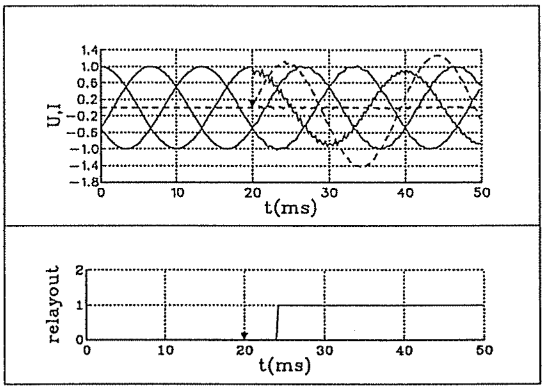

Figure 5 Fault at 64km

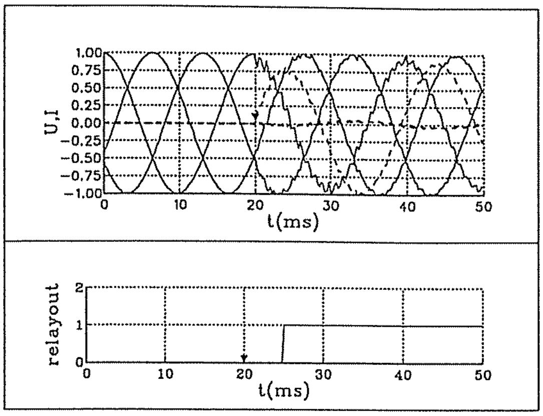

Figure 6 Fault at 100km

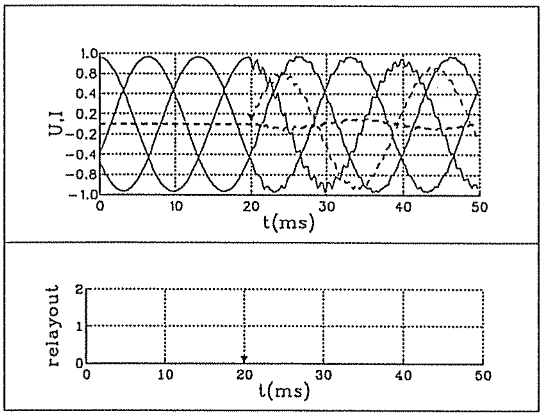

Figure 7 Fault at 110km

7. Conclusion

Based on a finite transform algorithm, a high speed digital distance protective relay has been developed. Fast discrete processing techniques are developed to specially tailored for the target 16-bit processor and the algorithm. Real-time simulation shows that all the relaying calculation can be completed within one sample interval with a very high sampling rate (4kHz).

A multi-processor relay structure has been developed with a separate analog to digital conversion board to deal with data acquisition. Dual port RAMs shared by two computer systems are proved to be very efficient. Real-time testing shows that the relay can operate at very high speed with high reliability. The relay can operate within 5ms in most situations.

Acknowledgments

The authors would like to thank the Hong Kong Polytechnic for the electrical engineering laboratory and computing facilities, and the provision of grants to make this project possible. Thanks also go to the Dept. of Electrical Power Engineering & Automation, Tianjin University for technical support.

References

1. Hicks K. L., Butt W. H., "Feasibility and Economics of Ultra-High-Speed Fault Clearing", IEEE Trans. on PAS, Vol. PAS-99, No.6, pp.2138-2143, 1980.

2. Phadke A. G., Hlibka T., and etc, "A Microcomputer Based Ultra-High-Speed Distance Relay: Field Tests", ibid, Vol. PAS-100, No.4, pp.2026-2033, 1981.

3. Johns A. T., Martin M. A., "New Ultra-High-Speed Distance Protection Using Finite-Transform Techniques", IEE Proc., Vol. 130, Pt. C, No.3, pp.127-138, May 1983.

4. Johns A. T., Martin M. A., and etc., "A New Approach to E. H. V. Direction Comparison Protection Using Digital Signal Processing Techniques", IEEE Trans. on PWRD, Vol. PWRD-1, No.2, pp.24-34, April 1986.

5. Prakash K. S., Malik O. P., Hope G. S., "Amplitude Comparator Based Algorithm for Directional Comparison Protection of Transmission Lines", ibid, Vol. PWRD-4, No.4, pp.2032-2041, October 1989.

6. Wiszniewski A., "Digital High-Speed Calculation of the Distorted Signal Fundamental Component", IEE Proc., Vol.137, Pt.C, No.1, pp.24, January 1990.

7. Li K. K., Cheung C., Xia Y. Q., "Processing Power System Signal for Digital Distance Protection", International Symposium on Signal Processing and its Applications, Australia, 1990, Conference Proceeding, Vol.2, pp.806-809.

8. Li K. K., Cheung C., Xia Y. Q., "A Modified Finite Transform Algorithm for Digital Distance Protection", Eighth CEPSI Conference, Singapore, 5-9 November 1990, Paper 4-41.

9. Johns A. T., Aggarwal R. K., "Digital Simulation of Faulted E. H. V. Transmission Lines with Particular Reference to Very-High-Speed Protection", ibid, Vol.123, No.4, pp.353-359, April 1976.

Appendix

$$[X_0<X(k)<0.8X_r]\wedge[R_0<R(k)<R_r+K_1X(k)]\wedge[X_m(k)>-X_0] \tag{34}$$

$$[0.8X_r<X(k)<X_r]\wedge[R_0<R(k)<R_r+K_1X(k)]\wedge[X_m(k)>-X_0] \tag{35}$$

where

$$K_1 = \mathrm{Rl1/Xl1}$$

$$X_m(k)-2\pi f[\bar{i}_{r1}(k)\bar{v}_{r2}(k-N_T)-\bar{i}_{r2}(k)\bar{v}_{r1}(k-N_T)]/D(k) \tag{36}$$

$X_m(k)$ is a measured reactance which arises through the need to directionalise the relay. It is calculated by utilizing voltage components sampled at N_T intervals previously and current component at any sample instant k. N_T is the number of samples in one cycle of power frequency.