LOW-FREQUENCY COMPUTER-BASED MULTIPHASE VARIABLE-AMPLITUDE VARIABLE-FREQUENCY OSCILLATOR

Z.O. Yonah,[1] M.M. Kissaka,[1] A.M. El-Serafi[2] and A.E. Krause[2]

1 Dept. of Electrical Eng. University of Dar es Salaam P.O. Box 35131 Dar es Salaam, Tanzania
2 Dept. of Electrical Eng. University of Saskatchewan, Saskatoon, Saskatchewan, Canada, S7N, 5A9

ABSTRACT

A computer-based oscillator scheme for generating low-frequency (0 - 100 Hz), symmetrical, multiphase variable-amplitude variable-frequency (VAVF) sinusoidal voltage signals is described. Real-time test results are presented for the cases of single-, two- and three-phase VAVF voltage signals. With this scheme, the amplitude, frequency, frequency range, number of phases and phase sequence of the signals can all be controlled dynamically in real time. In addition, the distortion of the generated voltage waveforms can be specified on-line and, once specified, it is independent of the frequency of the generated voltage signals.

INTRODUCTION

Controlled low frequency (0 - 100 *Hz*) multiphase variable-amplitude variable-frequency (VAVF) sinusoidal voltage signals are usually needed for (a) a.c. motor control schemes, e.g. in the control circuitry of cycloconverters and pulse-width modulated inverters, and (b) excitation schemes of asynchronized synchronous machines in variable-speed constant-frequency (VSCF) applications. In these applications, it is usually necessary to have a dynamic control of the amplitude, the frequency, and the phase sequence of the voltage control signals simultaneously. Such VAVF sinusoidal voltage control signals could be generated by various techniques [1-8]. These techniques are based either on a frequency-conversion approach in which a fixed frequency from a local oscillator is mixed with a variable frequency from a controllable frequency source [5] or on a frequency division method using digital combinational circuits [1 - 4]. In the frequency conversion approach, the mixer output is generally characterized by harmonic components having frequencies consisting of the sums and differences of the multiples of the variable input and fixed reference frequencies. Most of these frequency components are unwanted and, hence, low-pass filters are used to obtain the desired low frequency output from the mixer. This scheme of frequency synthesis applies to every phase generated. In this method, the phase information of each generated signal is expected to be preserved in the mixers. In the frequency division method, the output frequencies are synthesized by dividing a reference frequency by integer dividers which are programmed on digital counters. The desired waveforms are obtained by using sampled data of sine wave patterns which are converted into analog values by clocked digital-to-analog converters (DACs).

The above two frequency synthesis methods can be implemented by using different design techniques which can be classified into (a) analog designs, (b) digital designs, (c) hybrid designs, and (d) designs that use power converters as frequency changers [5, 8]. However, these designs have some drawbacks. For example, the analog and the hybrid designs suffer from the drift problem, aging of components, and environmental factors such as temperature. In addition, filtering the outputs is necessary to reduce the harmonic distortion of the waveforms. Hence, the design of such circuits requires careful selection of components to achieve the desired accuracy in the phase relationships and output frequencies. The main drawback of using power converters, e.g. cycloconverters, as frequency changers is that they are harmonic generators and have limited usable output frequency range. Usually this range is from d.c. to about a quarter of the supply frequency.

In a recent study [6], it has been shown that it is feasible to use a digital computer as a VAVF oscillator and this design offers two main advantages. Firstly, the technique is a software-based approach for generating VAVF sinusoidal waveforms of acceptable quality. Therefore, it offers a software-based flexibility for generating multiphase waveforms, which can be specified as mathematical functions, and for varying the number of control options. Secondly, the technique allows for the distortion of the waveforms generated to be specified on-line and, once specified, this distortion is independent of the output frequency. With this design method, no RC networks are required for phase shifting the waveforms to realize the phase relationships among the generated voltage signals. This design technique has the potential to overcome most of the drawbacks of the earlier design techniques. This paper describes the development, implementation, and testing of a multiphase oscillator scheme applying this computer-based technique.

THE OSCILLATOR ALGORITHM

The computer-based oscillator algorithm employs a playback dedicated D/A conversion process, which is driven by a real-time programmable clock, as a technique for generating the multiphase VAVF sinusoidal voltage signals each of the form $v_{\phi}(t) = V_p \sin(2\pi f_{da}t + \theta_0)$, where V_p is the peak amplitude, f_{da} is the frequency of the signals, and $0 \leq \theta_0 < 2\pi$ is the phase angle. The algorithm uses numerically computed samples of a sinusoidal waveform of specified amplitude as waveform pattern data. The waveform pattern is defined as: $P(\theta) = A\sin(\theta)$, where A is the specified amplitude and $0 \leq \theta < 2\pi$ radians. During the playback D/A conversion process, samples of the waveform pattern are fed to the DACs in a predetermined sequence and at controlled time intervals to generate the required sinusoidal voltage signals. This signal generation technique is illustrated in Fig. 1. In this figure, DRIVERs A, B, and C denote real-time clocked software programs. These drivers control the DACs identified as DAC#1, DAC#2, ... ,DAC#m. The timing of DRIVER A is controlled by CLOCK A and that of DRIVERs B and C is controlled by CLOCK B.

The function of DRIVER A is to synthesize and control the frequency f_{da} of the output signals $v_{\phi 1}(t)$, $v_{\phi 2}(t)$ ... $v_{\phi m}(t)$. The frequency f_{CK} of clock A is synthesized by DRIVER B as a function of a frequency command voltage signal $s_r(t)$. The pattern samples, U_1, U_2, ..., U_m are sent to the DACs at this clock rate, f_{CK}. The sign of $s_r(t)$, i.e. $sign[s_r]$, controls the phase sequence of the output voltages $v_{\phi 1}(t)$, $v_{\phi 2}(t)$ ... $v_{\phi m}(t)$. DRIVER C controls the amplitude of the output signals as a function of an amplitude command voltage signal $e(t)$ through the real-time multipliers denoted by X in Fig. 1. Also, it synthesizes the clock rate f_{sa} of

CLOCK B which controls the operations of DRIVERs B and C. The details of the functions of these drivers are discussed in the subsequent sections.

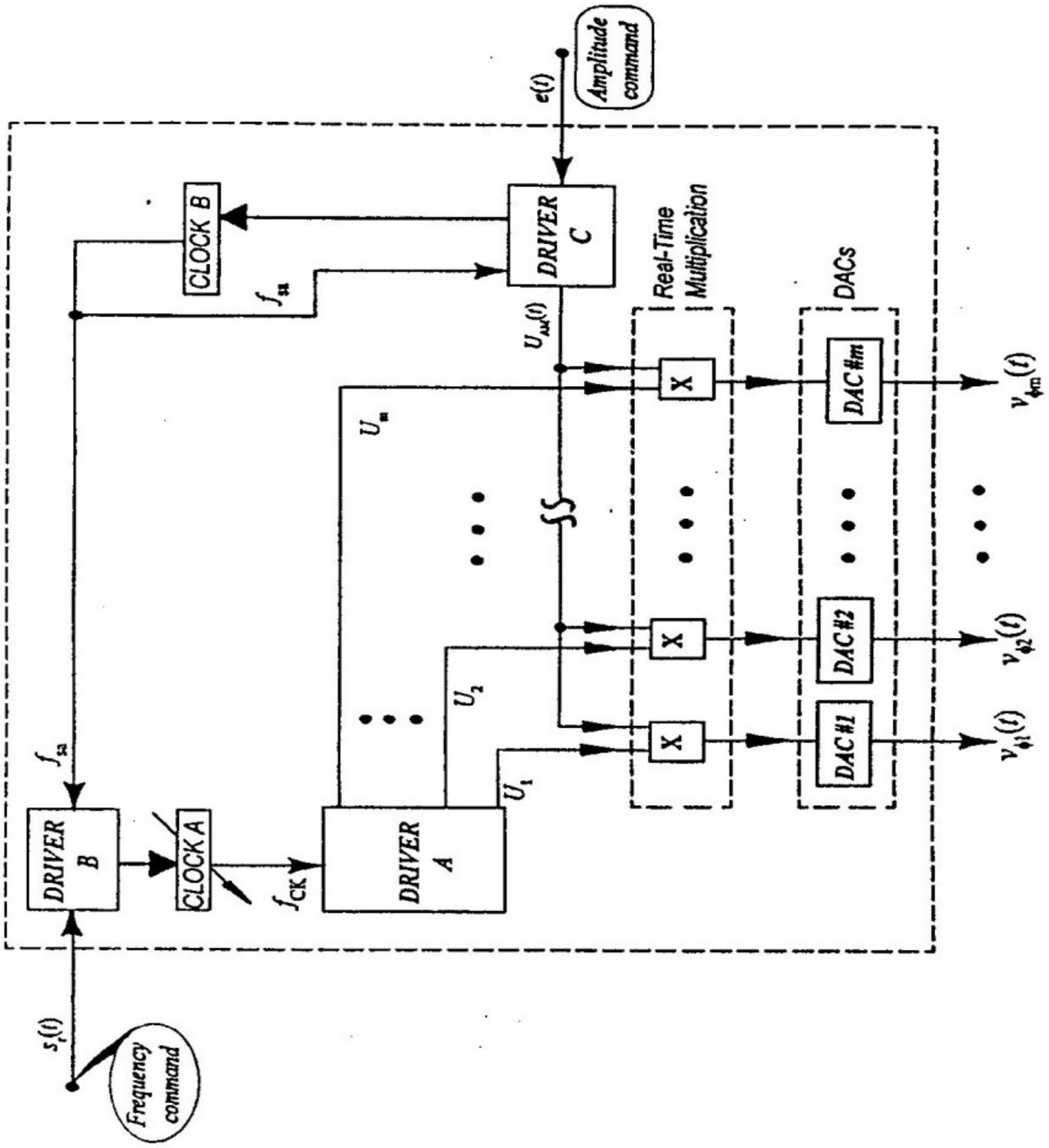

Fig. 1: A b;ock diagram showing the basic structure of the computer-based signal generation scheme.

FREQUENCY SYNTHESIS

Figure 2 shows typical periodic, stepped, sinusoidal waveform of the output voltage generated by a continuous D/A conversion process clocked at a constant rate f_{CK} for the case of $\theta_o = \pi/2$ radians. By inspecting this figure, an angle θ_n in radians can be associated with the stepped periodic signal at any instant of time t_n such that $\theta_n = 2\pi f_{da} t_n = (2\pi f_{da} n T_{CK})$ where $T_{CK} = 1/f_{CK}$ is the period of CLOCK A, and $t_n = nT_{CK}$, where n denotes the n^{th} clock pulse. If N is the number of samples per cycle of the wave pattern used in the D/A conversion process, the periodic output voltage will have N-steps of equal width and, thus, the period $T_{da} = 1/f_{da}$ of the output waveform will be given by $T_{da} = NT_{CK}$. This gives $f_{da} = f_{CK}/N$ and $\theta_n = 2\pi n/N$. The relationship $f_{da} = f_{CK}/N$ shows that the frequency of the output voltages, $v_{\phi 1}(t)$, $v_{\phi 2}(t)$ ... $v_{\phi m}(t)$ can be varied by dynamically varying either the number of samples per cycle N, the sampling frequency f_{CK} of DRIVER A or both. Furthermore, for $n = 0, 1, 2, ..., N - 1$, the samples of each generated phase voltage can be expressed as:

$$P(\theta_n) = A\sin(2\pi(n + n_O)/N) \tag{1}$$

where $\theta_o = 2\pi n_O/N$. From Eqn.(1), the N sets of ordinates (P_n, $n = 0, 1, 2, ..., N - 1$) of the wave pattern are independent of the frequency to be generated. Thus, these values can be computed off-line and stored in the computer memory as a pattern lookup table.

In this oscillator scheme, the variable N represents contiguous memory locations where the pattern data are stored. For this reason, the values of N can only be integer values. Consequently, the major source of error in the oscillator scheme is the round-off errors or quantization noise due to the finite number of steps (= N) within a cycle of each output waveform [6]. The number of samples per cycle N should be chosen such that the total waveform distortion of the stepped signals will not exceed the maximum specified level. Usually, the maximum acceptable distortion level is 5% [5]. Given a value of N, the value of the voltage distortion

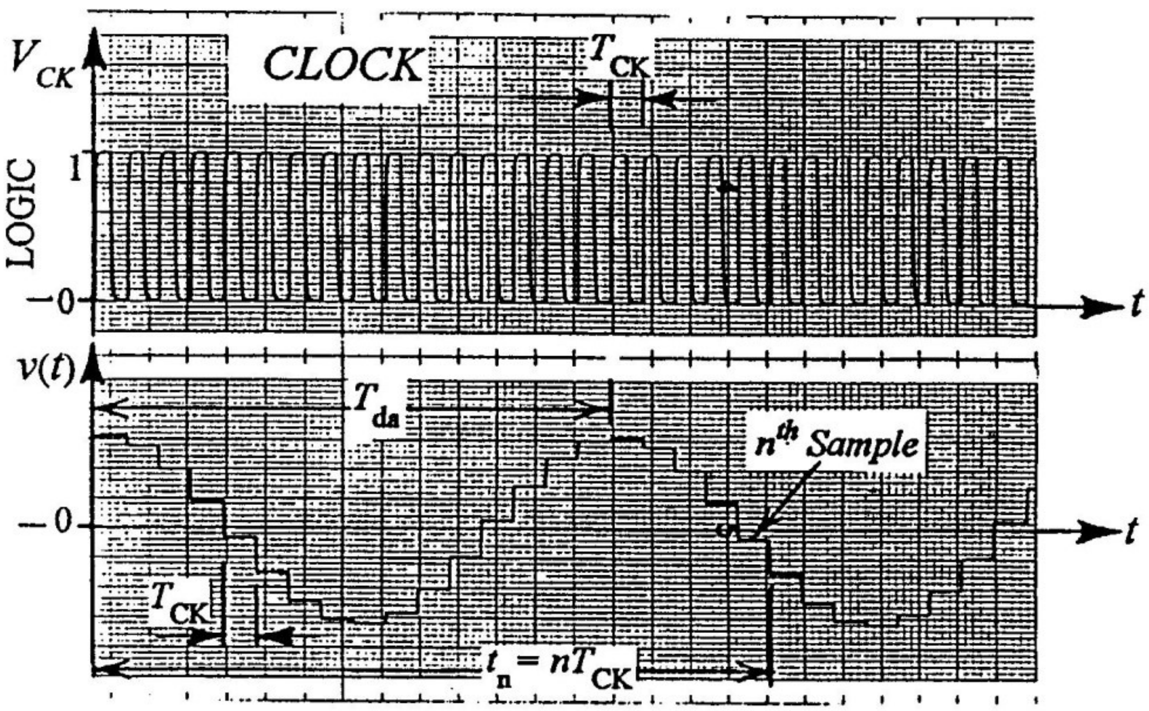

Fig. 2: **A periodic-stepped output voltage from a digital-to-analog converter driven by a clocked software driver.**

factor (VDF) for such stepped waveforms can be determined from $VDF\% = 180/N$ [6]. The harmonics responsible for the waveform distortion are integer multiples of the fundamental frequency, f_{da}, and can be identified by the expression $(kN \pm 1) f_{da}$ [2, 6] where k is a positive integer. It is therefore preferable to select and keep the value of N larger than a certain minimum value. For example, choosing $N > 180$ samples per cycle gives a distortion factor of less than one percent and, thus, it may not be necessary to perform low-pass filtering to the generated signals.

FREQUENCY CONTROL

Control of the output frequency, f_{da}, is accomplished by the use of the real-time frequency command voltage signal $s_r(t)$. This is achieved by associating a fixed frequency f_0 to the angle θ_n such that $\theta_n = 2\pi\, s_n\, f_0\, t_n = sign[s_r]2\pi\, s_{ra}\ f_0\, t_n$ where $s_{ra} = |\, s_r\,|$. This gives $f_{da} = s_a\ f_0$, where $sign[s_r]$ and s_{ra} are respectively the sign and magnitude of $s_r(t)$ at time $t_n = nT_{CK}$ (Fig. 2). The equation $f_{da} = s_{ra}\, f_0$ defines a frequency modulation scheme in which the signal $s_r(t)$ modulates the reference frequency f_0 to obtain the output frequency f_{da}.

DYNAMIC CONTROL OF THE NUMBER OF PATTERN SAMPLES, NUMBER OF PHASES, AND PHASE SEQUENCE

DRIVER A is programmed to initialize a pattern lookup table which holds a specified maximum number, N_M, of equal-spaced samples per cycle of the pattern wave. The angular spacing between the consecutive samples of the waveform is $2\pi/N_M$. DRIVER A has an interactive software pointer control process for scanning the lookup table of pattern data. The pointer control process generates sample indices n_i where $k = i + 1$ denotes the k^{th} phase ($i = 0, 1, .., m - 1$) for accessing the pattern samples to generate the data streams U_1, U_2, ..., U_m (Fig. 1). In other words, there is a separate index n_i for each phase k which selects appropriate pattern data and assigns the data as U_k which is then used to generate the phase voltage signal $v_{\phi k}(t)$. During the signal generation process, the pointer scanning process must update the sample indices n_i such that:

(1) The value of N remains fixed as specified,
(2) correct number of phases are generated,
(3) the phase sequence of the output signals corresponds to $sign[s_r]$ correctly, and
(4) continuous (or seamless) signal waveforms are generated, i.e. no discontinuities.

The values of N are derived dynamically by the software pointer control process as it scans the N_M coordinates of the wave pattern. The scanning is such that $N = N_M/J$, where $J = 1, 2, ..., N_M - 1$. The integer J is a jump index. For example, $J = 1$ will give $N = N_M$. In this case, every sample in the pattern will be used for the signal generation. When $J = 2$, every other sample in the pattern sequence will be used. This will give $N = N_M /2$. The oscillator scheme allows the value of N to be specified interactively. Conditions (1), (2), and (3) of the pointer scanning process are satisfied by updating the indices n_i at every clock pulse to:

$$n_i \text{ <-------- } n_i + sign[s_r].J - i\mu \tag{2}$$

where $\mu = 0$ for $m = 1$, $\mu = N_M /4$ for $m = 2$, and $\mu = N_M /m$ for $m \geq 3$, and $i = 0, 1, ..., m - 1$. These settings will give exact values of phase shift if N_M is selected to be a multiple of N. In order to realize continuous signal waveforms (condition (4) of the pointer scanning process), each sample index n_i is reset to an appropriate value when it reaches a value greater than N_M - 1 or less than zero. The pseudo code for resetting n_i is:

$$n_i \text{ <-------- } n_i - sign[s_r].N_M \tag{3}$$

These two pseudocodes (2) and (3) describe the pointer scanning process as a clocked, resettable, software-based digital oscillator as illustrated in Fig. 3. In this figure, the software-based switch S_1 is a boolean logic that compares the value of each index n_i with that of N_M and selects the appropriate bias addend for the oscillator. Also, the switch S_2 is another boolean logic that compares i with m. The values of n_i and i are stored in memory registers *MEM1* and *MEM2*, respectively. The two boolean logics provide the oscillator with a reversible ring structure of the pattern data as well as a multiphase structure. In real time, the data appear continuous or seamless. The two pseudo codes use the sign of the frequency command signal, $sign[s_r]$, to select the direction of scanning, i.e. incrementing n_i for $sign[s_r] = +1$, decrementing n_i for $sign[s_r] = -1$ and keeping n_i a constant for $sign[s_r] = 0$. Effectively, this is equivalent to controlling the phase sequence of the signal voltages.

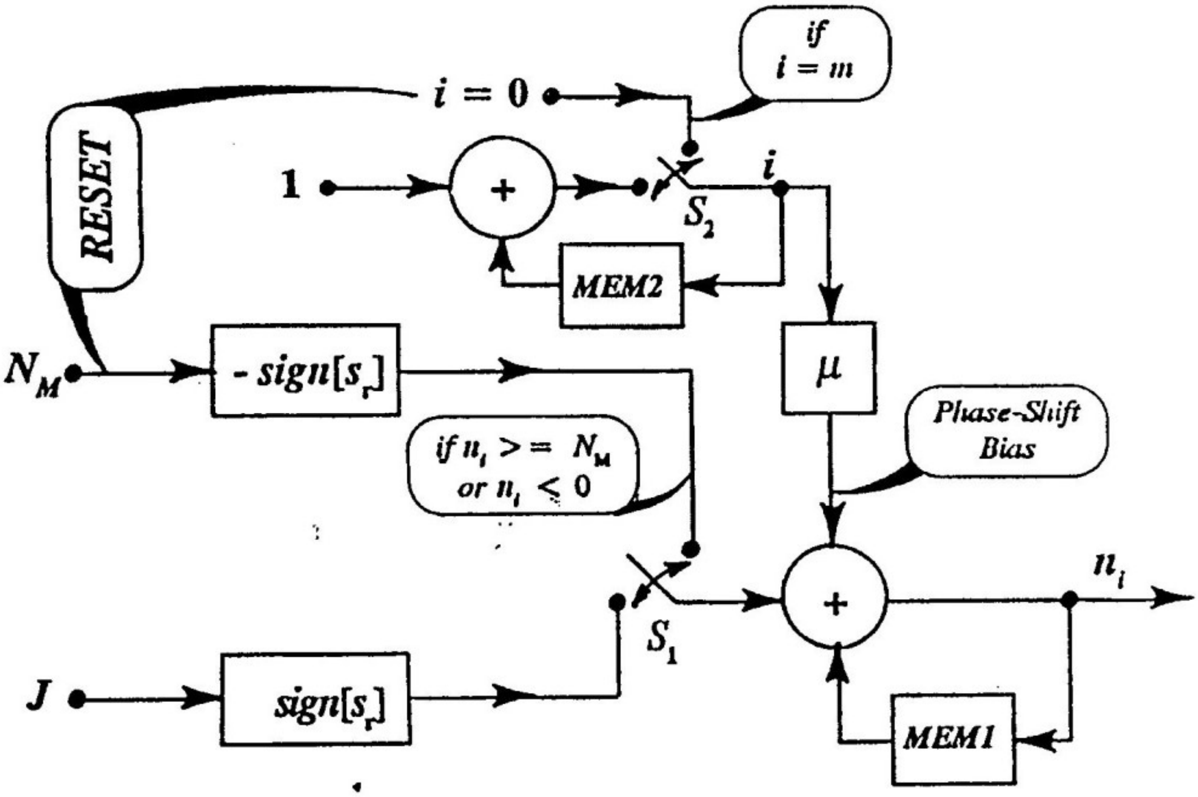

Fig. 3: A block diagram of a clocked, resettable, software-based digital oscillator.

Since the waveform distortion of the generated voltage signals is a function of N, the pointer control process can be used not only to vary the frequency, but also to set the distortion of the waveforms dynamically. In the case that the clock frequency f_{CK} is varied to control the output frequency f_{da}, this pointer control process can be used to change the frequency range of the output frequencies by varying J dynamically, which in turn varies N.

AMPLITUDE CONTROL

The adjustment of the amplitude V_p of the output voltages $v_{\phi 1}(t)$, $v_{\phi 2}(t), \ldots, v_{\phi m}(t)$ is the task of DRIVER C in Fig.1. This driver generates a signal $U_{AM} = U_{AM}(t)$ which is a function of the amplitude command signal $e(t)$. Then, this signal is multiplied by the outputs of DRIVER A, i.e. $U_1, U_2, \ldots, U_m$, such that $V_p = A\,U_{AM}$. The oscillator scheme has two amplitude control functions, i.e.

$$V_{pn} = \alpha e_n \tag{4}$$
$$V_{pn} = V_0(1 \pm e_n) \tag{5}$$

where α is a proportionality factor, V_0 is a reference (bias) amplitude, and V_p and e_n are the processed updates of V_p and $e(t)$ corresponding to the n^{th} clock pulse, respectively. Equation (4) describes an attenuation or amplification command and Eqn. (5) describes a positive (+) or a negative (-) amplitude modulation command. The negative amplitude modulation case allows the oscillator to be used in a closed loop feedback control system in which $e(t)$ becomes the error signal. Since the control functions are software-based, any other suitable amplitude control function can be easily implemented. The multiplications in Eqns. (4) and (5) is best realized through hardware multiplication by using multiplying DACs [8].

IMPLEMENTATION

Figure 4 shows a block diagram of the implemented computer-based oscillator for generating up to three-phase VAVF sinusoidal voltage signals. In this figure, PC1, PC2, and PC3 are 8086 - AT&T/10 *MHz* - microcomputers with bus speeds of 5 *MHz*. These three microcomputers are programmed as dedicated modules for signal generation, frequency control, and amplitude control, respectively. The three PCs execute the three real-time application programs, DRIVERs A, B, and C as illustrated in Figs.1 and 4. The three-PC setup has been used for the sole reason of simplifying the implementation of the scheme of Fig. 1 for research purposes. All the three drivers may be implemented on one PC. Plug-in cards are used to realize the required input/output (I/O) interfaces. The analog-to-digital converters (ADCs), timers and the digital I/O interfaces are provided on DAS-8 cards. The digital-to-analog converters (DACs) are 12-*bit* multiplying DACs provided on DAC-02 cards [9 - 11].

Figure 4 also shows a bidirectional, programmable, 24-*bit* wide parallel data line connecting PC1 and PC2. This data line is configured using two PIO-12 plug-in cards, one on each of the two PCs. The 8255 programmable peripheral interface IC on each of the PIO-12 cards provides the means to control the 24 bits of the digital I/O data line by software [10]. The microcomputer PC1 receives coded data containing the phase sequence ($sign[s_r]$) and the value of the jump-index, J, from PC2 via the data line. The value of J is used in PC1 to select a data pattern that defines one cycle of the sine wave pattern, i.e. the number of samples per cycle, N. The selected data is then used to generate the voltage signals using DAC_{11}, DAC_{12}, and DAC_{13}.

The pattern data (real numbers) have to be converted into integer values for loading into the latches of the DACs. The range of the integer values, D, should be $0 \leq D \leq 2^b - 1$, where *b-bits* is the width-size of the data format of the DACs used. For the purpose of computing the values of D, a reference supply voltage U_0 (not shown in Fig. 1) has to be specified such that when the DACs are supplied with this value, the generated signals will have the specified amplitude, A. In general, the digital integer data D corresponding to a pattern ordinate U_k, $k = 1, 2, 3$ (Fig. 1), of a bipolar sine wave is given by:

$$D = INT(2^{b-1}(1 + U_k / U_0) + 0.5);\ \text{by rounding} \qquad (6)$$

For $U_0 = -5\,V$, $b = 12$-*bits*, using the DACs on the DAC-02 cards, Eqn. (6) becomes:

$$D = INT(2048.5 - 409.6U_k);\ \text{by rounding} \qquad (7)$$

The sample rate f_{CK} of application of the pattern data to the DACs of PC1 is controlled by a square wave signal from PC2 which is fed through a digital I/O port in PC1. This square wave signal, CLOCK A, is

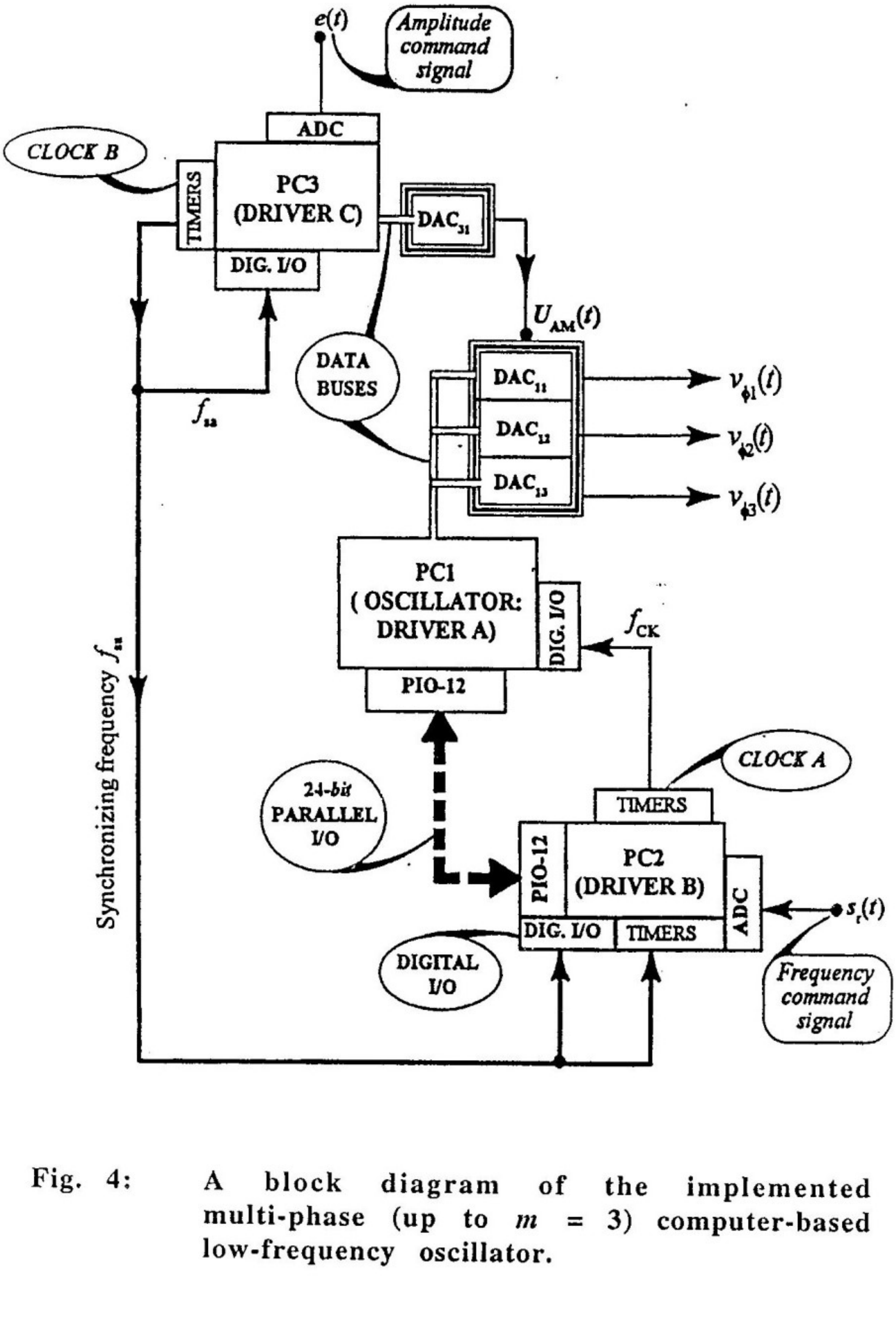

Fig. 4: A block diagram of the implemented multi-phase (up to m = 3) computer-based low-frequency oscillator.

generated by programmable counters in PC2 (Fig 4). The frequency f_{CK} of the CLOCK A signal is adjusted according to:

$$f_{\text{CK}} = s_{\text{ra}} f_0 N \tag{8}$$

where f_0 is the reference frequency. Another square wave signal generated in PC3 serves as CLOCK B signal. This signal has a frequency f_{sa} and is used to synchronize the sampling operations in PC2 and PC3. In Fig. 4, the real-time multiplication of Fig. 1 is implemented by having the reference supply U_0 of DAC$_{11}$, DAC$_{12}$, and DAC$_{13}$ controlled by the output U_{AM} from DAC$_{31}$ of PC3.

RESULTS

Two signal generators were used to supply the frequency command signal, $s_r(t)$, and the amplitude command signal, $e(t)$. The signal $s_r(t)$ was sampled by using the ADC of PC2 and $e(t)$ was sampled by using the ADC of PC3 (Fig. 4). Both test signals were sampled at $f_{\text{sa}} = 30\ Hz$. This rate was investigated and found to be adequate for testing the oscillator.

Frequency synthesis by using a programmable real-time clock is illustrated in Fig. 5. In this figure, low frequencies ($\leq 10\ Hz$) for clock (A) were selected so that the clock signal, V_{CK}, could be recorded. Evident from this figure is that increasing the clock rate f_{CK} for constant number of samples per cycle, N, increases the frequency f_{da} of the generated voltage signals. With this frequency scheme, low frequencies ($< 100\ Hz$) can be generated with a high accuracy in the order of $10^{-3}\ Hz$.

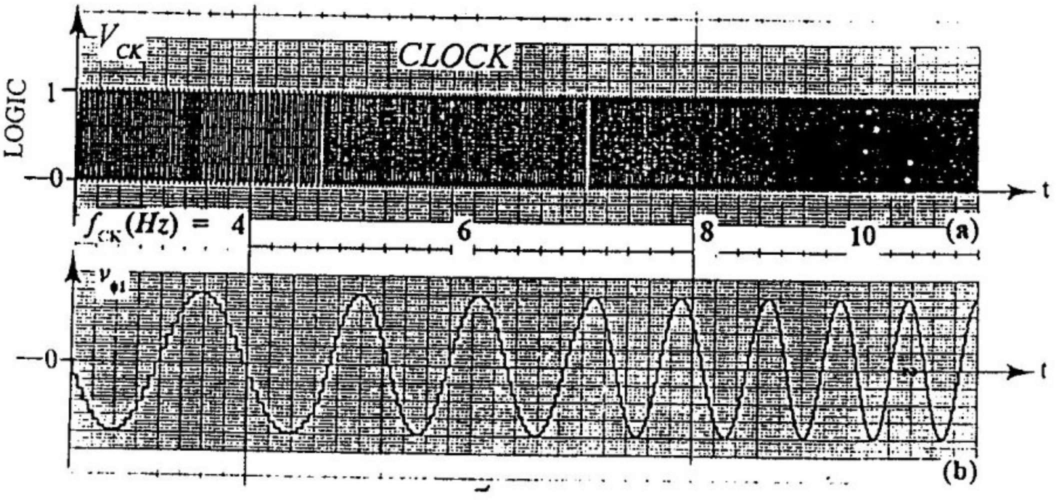

Fig. 5: **Response to changes in the clock rate f_{CK} for constant number of samples per cycles (N = 32). Chart sensitivity: Horizontal - 0.2 *sec/div.*, Vert. - 200 *mV/div.* for the clock signal (V_{ck}) and 100 *mV/div.* for v_d**

Figure 6 shows the effects of changing the number of samples per cycle, N, at a constant frequency for CLOCK A, f_{CK}, on the generated two-phase voltage signals ($v_{\phi 1}(t)$ and $v_{\phi 2}(t)$). One of these effects is that the size of the steps of the waveforms of the two signals decreases as N increases and, thus, the waveforms become smoother with increasing N. The second effect is that increasing N at a constant clock frequency decreases the frequency of the generated signals, f_{da}. In addition, Fig. 6 shows that the oscillator scheme can respond to changes in N dynamically and, thus, the frequency range of the generated signals can be varied dynamically.

Figures 7, 8, and 9 show the VAVF voltage signals generated as a result of the oscillator response to the frequency and amplitude command signals. Figures 7 and 8 show results for the case of two-phase voltage signals and Fig. 9 shows the three-phase voltage signals. The sign of $s_r(t)$, i.e. $sign[s_r]$ should trigger the change of phase sequence of the generated signals. This change in the phase sequence can be observed on the output waveforms at the instants of zero crossings of the signal $s_r(t)$.

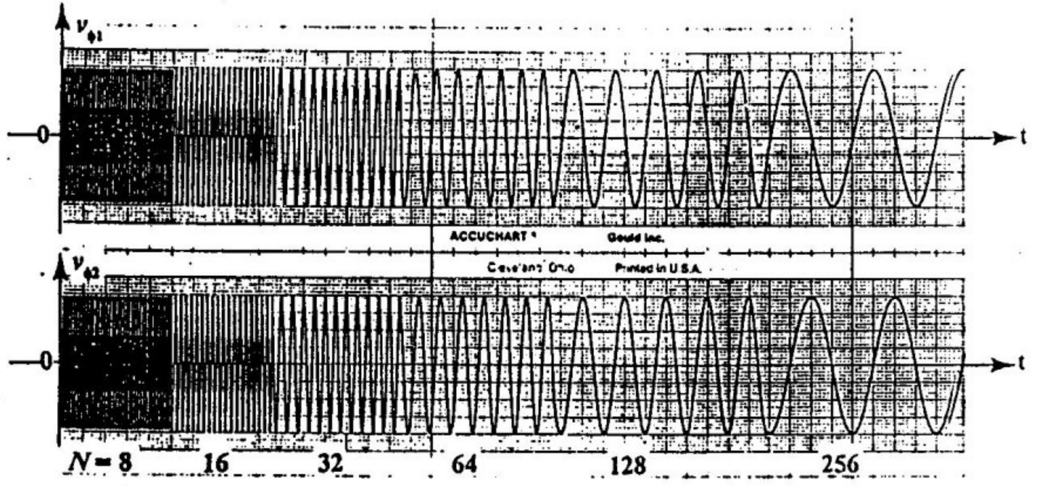

Fig. 6: **Response to changes in the number of samples per cycle, N, at constant clock rate f_{CK} = 64 Hz). Chart sensitivity: Horizontal - 0.2 *sec/div.*, Vert. - 100 *mV/div.***

For all the three figures, the frequency of the output signals is given by $f_{da} = | s_r | f_0$, where $f_0 = 60$ Hz. In Fig. 7, the amplitude of the output signals varies according to Eqn. (5) where $V_0 = 1.0$ V, and $V_p = (1 - e)$ *Volts*. In Figs. 8 and 9, $V_p = 0.4e$ *Volts* (Eqn. (4)). It is clear from Figs. 7 to 9 that the two-phase case results in two orthogonal outputs $v_{\phi 1}(t)$ and $v_{\phi 2}(t)$ and the three-phase case results in outputs $v_{\phi 1}(t)$, $v_{\phi 2}(t)$ and $v_{\phi 3}(t)$ which are 120° phase shifted from each other. The results of these figures show also that the oscillator scheme can modulate both the amplitude and frequency of the output signals accurately. Also, from these figures, it can be noticed that the tracking of the signals $s_r(t)$ and $e(t)$ is fast and accurate.

CONCLUSIONS

This paper has described a computer-based oscillator scheme for generating low-frequency (0 - 100 Hz), multiphase, variable-amplitude variable-frequency (VAVF) sinusoidal voltage signals. This scheme is a flexible signal generator in that the amplitude, frequency, frequency-range, number of phases to be generated, and phase sequence of the output voltage signals can all be controlled dynamically. The oscillator scheme can be implemented by using standard plug-in data

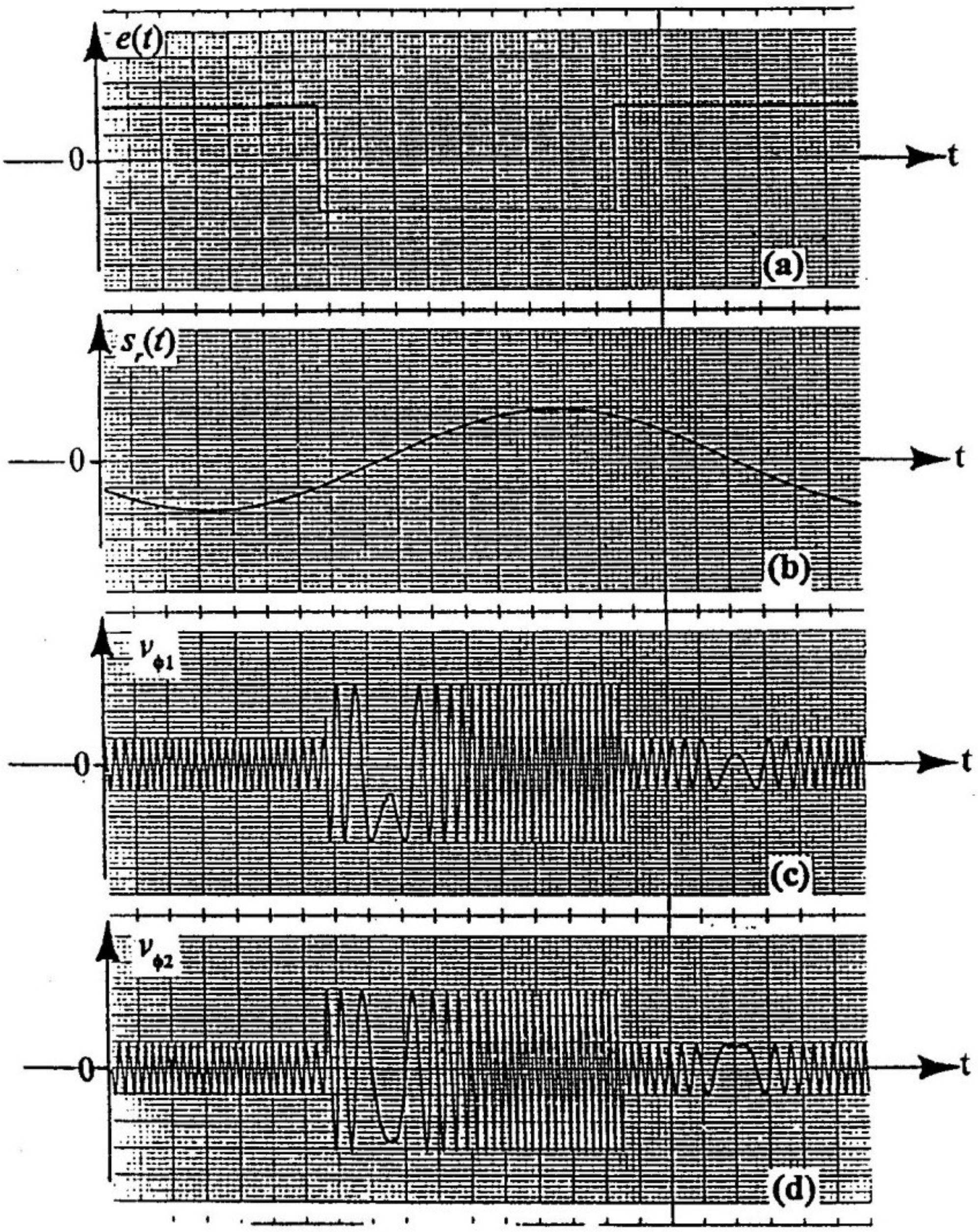

Fig. 7: **Two-phase VAVF voltage signals as oscillator response to frequency and amplitude command signals (N= 128).** CASE: Negative - amplitude modulation. Chart sensitivity: Horizontal - 0.2 *sec/div.*, Vertical- (a) 50 *mV/div.*, (b) 1*%/div.*, (c) and (d) 100 *mV/div.*

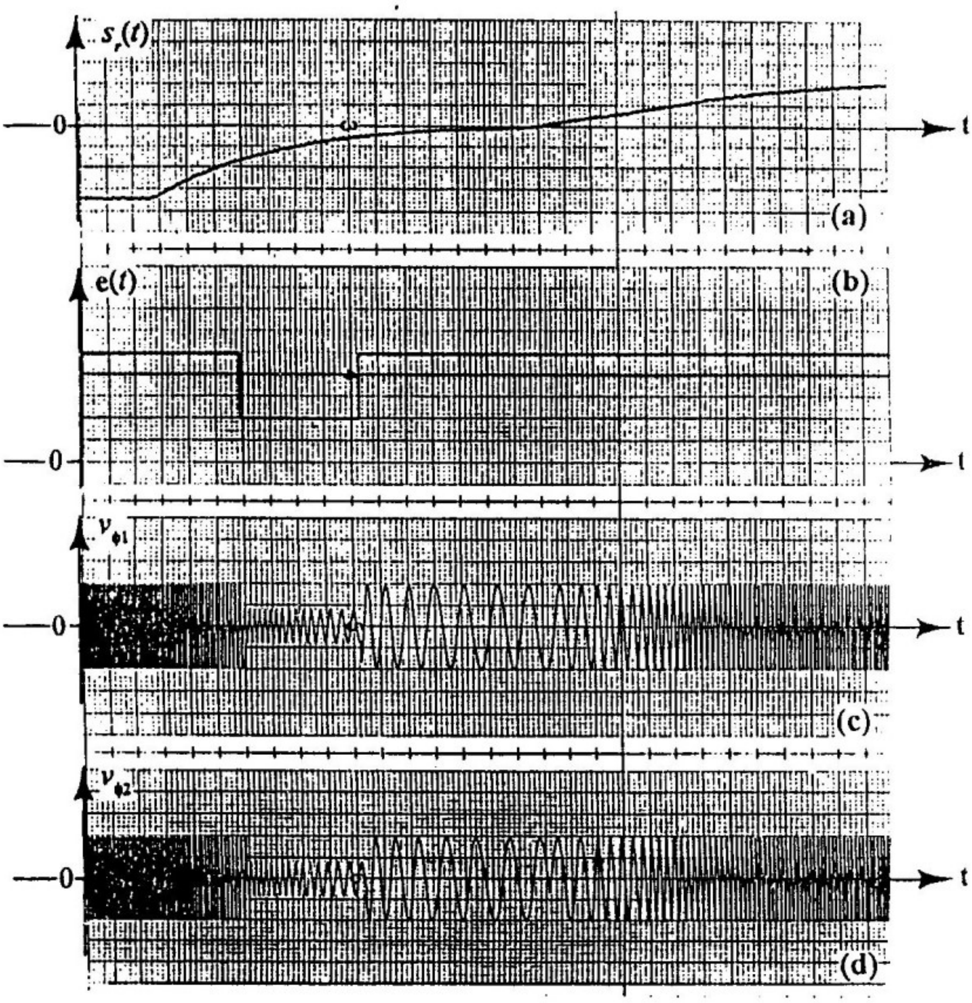

Fig. 8: **Two-phase VAVF voltage signals as oscillator response to frequency and amplitude command signals ($N = 128$).** CASE: Proportional - amplitude control. Chart sensitivity: Horizontal - 0.2 *sec/div.*,Vertical- (a) 1*%/div.*, (b), (c) and (d) 200 *mV/div.*

acquisition cards and standard personal computers (PCs). The use of such plug-in cards allows for the oscillator scheme to be used as a data acquisition system, a personal instrument for metering purposes, or/and a controller. The implemented oscillator is currently being used as a two-phase excitation system for a variable-speed dual-excited synchronous generator used as a variable-speed constant-frequency (VSCF) generating system [7, 8].

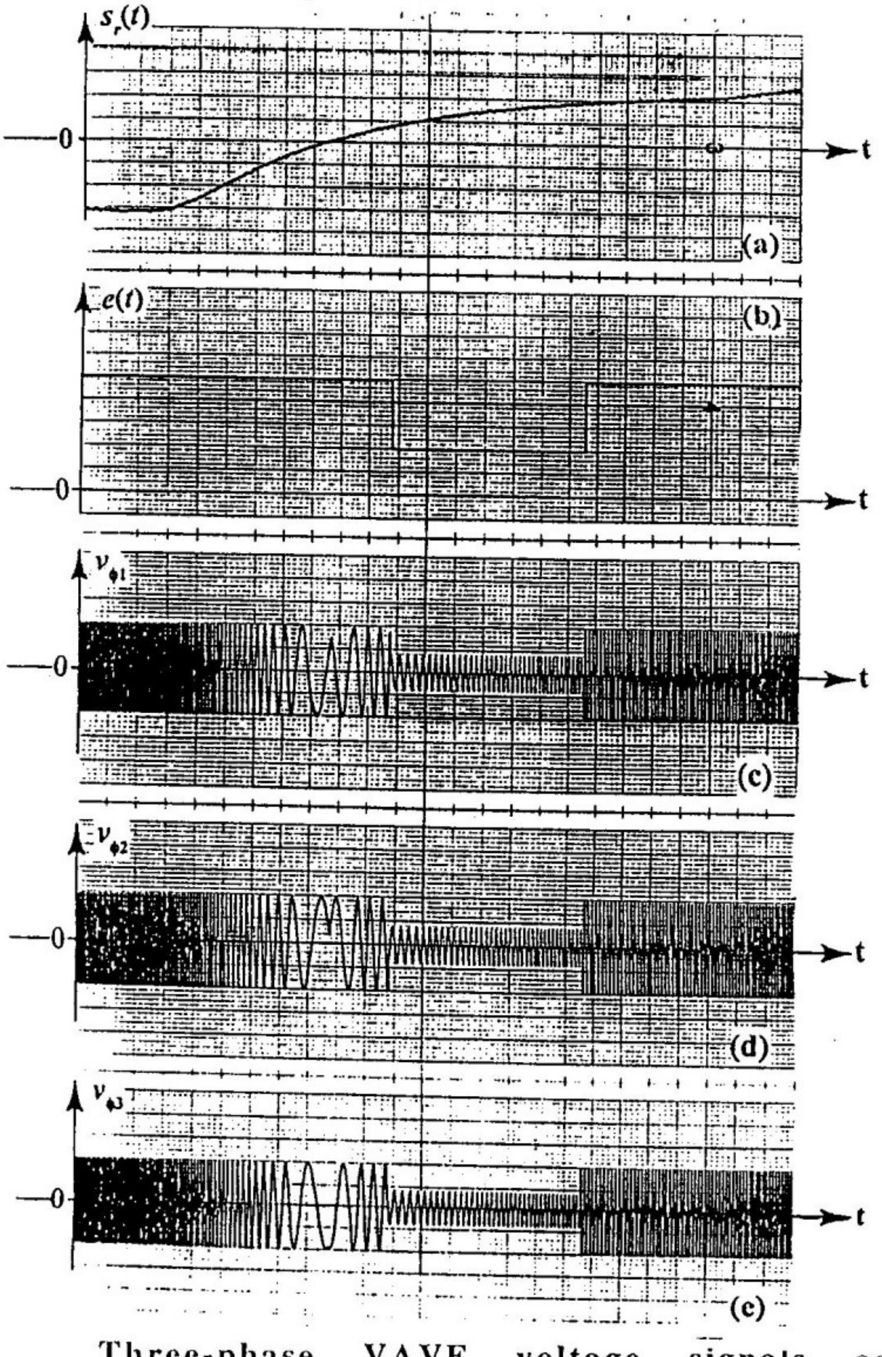

Fig. 9 **Three-phase VAVF voltage signals as oscillator response to frequency and amplitude command signals (N= 128).** CASE: Proportional - amplitude control. Chart sensitivity: Horizontal - 0.2 *sec/div.* , Vertical- (a) 1*%/div.*, (b), (c) (d) and (e) 200 *mV/div.*

REFERENCES

1. J. Tierney, C.M. Rader, and B. Gold, A Digital Frequency Synthesizer,*IEEE Transactions on Audio and Electroacoustics,* Vol. AU-19, 1971, pp. 48-57.
2. R.W. Burhans, Low-Bit Sine-Wave Approximations for Audio Signal Sources, *Journal of Audio Engineering Society,* Vol. 21, 1973, pp. 277-279.
3. J. Gorski-Popiel, Editor, *Frequency Synthesis: Techniques and Applications*, IEEE Press, *1975.*
4. H,G. Alles, Music Synthesis Using Real Time Digital Techniques, *Proceedings of IEEE,* Vol 68, 1980, pp. 436-449.
5. D.G. Fink, and D. Christiansen, Editors, *Electronics Engineers' Handbook,* Mcgraw-Hill, 1989.
6. Z.O. Yonah, A.M. El-Serafi, and A.E. Krause, On the Use of a Digital Computer as a Variable-Amplitude Variable-Frequency Oscillator, *IEEE Transactions on Industrial Electronics,* Vol. IE-37, 1990, pp. 576-581.
7. Z.O. Yonah, A.M. El-Serafi, and A.E. Krause, Regulation of Terminal Voltage and Output Frequency of a Variable-Speed Dual-Excited Synchronous Generator by Using a Computer-Based Two-Phase Excitation System, *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, Vancouver, British Columbia, Canada, September 14-17, 1993, pp. 1065 - 1068.
8. Z.O Yonah, Microcomputer-Based Real-time Two-Phase Excitation System for a Variable-Speed Dual-Excited Synchronous Generator, PhD Thesis, University of Saskatchewan, Canada, August, 1994.
9. Advanced Micro Devices,Inc., *MOS/LSI Data Book*, AM-PUB118, 1980.
10. Intel Co., *Peripherals-1990 Intel Data Handbook,* Intel Corporation, 1990.
11. Omega Engineering, Inc., *The Data Acquisition Systems Handbook-Vol. 27*, Omega Engineering, Inc., 1990.