Scientific Research Publishing

Journal of Applied Mathematics and Physics, 2023, 11, 2195-2204
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

Quantum Computer on Nvidia GPU

Alexander Soiguine

SOiGUINE Quantum Computing, Aliso Viejo, USA
Email: alex@soiguine.com

How to cite this paper: Soiguine, A. (2023) Quantum Computer on Nvidia GPU. *Journal of Applied Mathematics and Physics*, **11**, 2195-2204.
https://doi.org/10.4236/jamp.2023.118141

Received: July 13, 2023
Accepted: August 11, 2023
Published: August 14, 2023

Abstract

Geometric Algebra formalism opens the door to developing a theory deeper than conventional quantum mechanics. Generalizations, stemming from implementation of complex numbers as geometrically feasible objects in three dimensions, unambiguous definition of states, observables, measurements, Maxwell equations solution in those terms, bring into reality a kind of physical fields spreading through the whole three-dimensional space and values of the time parameter. The fields can be modified instantly in all points of space and time values, thus eliminating the concept of cause and effect, and perceiving of one-directional time. In the suggested theory all measured observable values get available all together, not through looking one by one. In this way quantum computer appeared to be a kind of analog computer keeping and instantly processing information by and on sets of objects possessing an infinite number of degrees of freedom. As practical implementation, the multithread GPUs bearing the CUDA language functionality allow to simultaneously calculate observable measurement values at a number of space/time discrete points only restricted by the GPU threads capacity.

Keywords

Geometric Algebra, Quantum Mechanics, Wave Functions, Maxwell Equations, GPU, CUDA

1. Introduction. What a Legacy Analog Computer Is

An analog computer is generally a type of computing device that uses the continuous variation aspect of physical phenomena to model the problem being solved [1].

One of the most effective of such kind of relations is similarity between linear m echanical components and electrical components since they can be modeled using equations of the same form. Electronic analog computers were often used when a system of differential equations proved very difficult to solve by

traditional means. As an example, the dynamics of a spring-mass system can be described by the equation $m\ddot{y}+d\dot{y}+cy=mg$, or $\ddot{y}=-\frac{d}{m}\dot{y}-\frac{c}{m}y-g$ with y as the vertical position of a mass m, d the damping coefficient, c the spring constant and g the gravity of Earth. The equivalent analog circuit consists of two integrators, Figure 1, for the state variables $-\dot{y}$ (speed) and y (position), one inverter, Figure 2, and three potentiometers.

2. Geometric Algebra Type of Analog Modeling Computer

Different physical phenomena to model problems is considered below.

The circular polarized electromagnetic waves are the type of waves following from the solution of Maxwell equations in free space done in geometric algebra terms [2] [3]. Let us take the electromagnetic field in the form:

$$F=F_0\exp\left[I_S\left(\omega t-k\cdot r\right)\right] \tag{1}$$

which should be solution of

$$\left(\partial_t+\nabla\right)F=0 \tag{2}$$

Solution of (2) should be sum of a vector (electric field e) and bivector (magnetic field I_3h):

$$F=e+I_3h$$

with some initial conditions:

$$e+I_3h\big|_{t=0,\vec{r}=0}=F_0=e\big|_{t=0,\vec{r}=0}+I_3h\big|_{t=0,\vec{r}=0}=e_0+I_3h_0$$

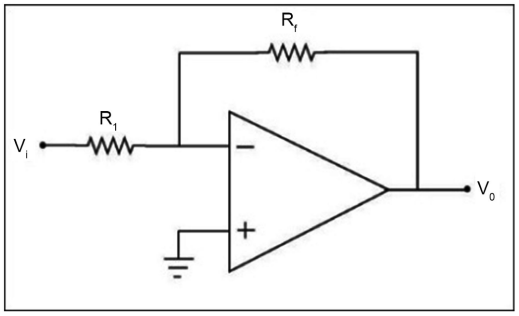

Figure 1. Integrator.

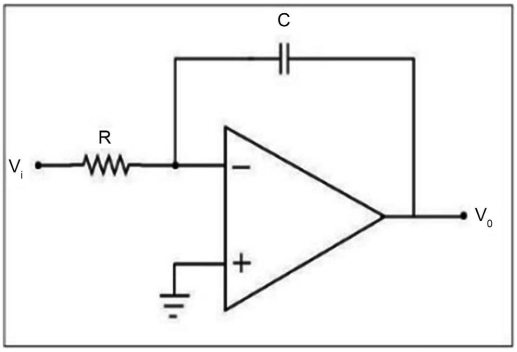

Figure 2. Inverter.

For given plane S in the (1) solution of the three-dimensions Maxwell Equation (2) has two options

- $F_+ = (e_0 + I_3 h_0)\exp\left[I_S(\omega t - k_+ \cdot r)\right]$, with $\hat{k}_+ = I_3 I_S$, $\hat{e}\hat{h}\hat{k}_+ = I_3$, and the triple $\left\{\hat{e}, \hat{h}, \hat{k}_+\right\}$ is right hand screw oriented, that's rotation of $\hat{e}$ to $\hat{h}$ by $\pi/2$ gives movement of *right hand screw* in the direction of $k_+ = |k| I_3 I_S$.
- $F_- = (e_0 + I_3 h_0)\exp\left[I_S(\omega t - k_- \cdot r)\right]$, with $\hat{k}_- = -I_3 I_S$, $\hat{e}\hat{h}\hat{k}_- = -I_3$, and the triple $\left\{\hat{e}, \hat{h}, \hat{k}_-\right\}$ is left hand screw oriented, that's rotation of $\hat{e}$ to $\hat{h}$ by $\pi/2$ gives movement of *left hand screw* in the direction of $k_- = -|k| I_3 I_S$ or, equivalently, movement of *right hand screw* in the opposite direction, $-k_-$.
- e_0 and h_0, initial values of e and h, are arbitrary mutually orthogonal vectors of equal length, lying on the plane S. Vectors $k_\pm = \pm|k_\pm| I_3 I_S$ are normal to that plane. The length of the "wave vectors" $|k_\pm|$ is equal to angular frequency ω.

Maxwell Equation (4.2) is a linear one. Then any linear combination of F_+ and F_- saving the structure of (1) will also be a solution.

Let's write:

$$\begin{cases} F_+ = (e_0 + I_3 h_0)\exp\left[I_S \omega\left(t - (I_3 I_S)\cdot r\right)\right] = (e_0 + I_3 h_0)\exp\left[I_S \omega t\right]\exp\left[-I_S\left[(I_3 I_S)\cdot r\right]\right] \\ F_- = (e_0 + I_3 h_0)\exp\left[I_S \omega\left(t + (I_3 I_S)\cdot r\right)\right] = (e_0 + I_3 h_0)\exp\left[I_S \omega t\right]\exp\left[I_S\left[(I_3 I_S)\cdot r\right]\right] \end{cases} \quad (3)$$

Then for arbitrary (real[1]) scalars λ and μ:

$$\lambda F_+ + \mu F_- = (e_0 + I_3 h_0)\mathrm{e}^{I_S \omega t}\left(\lambda \mathrm{e}^{-I_S\left[(I_3 I_S)\cdot r\right]} + \mu \mathrm{e}^{I_S\left[(I_3 I_S)\cdot r\right]}\right) \quad (4)$$

is solution of (4.2). The item in the second parenthesis is weighted linear combination of two states with the same phase in the same plane but opposite sense of orientation. The states are strictly coupled, entangled if you prefer, because bivector plane should be the same for both, does not matter what happens with that plane.

Arbitrary linear combination (4.4) can be rewritten as:

$$\lambda \mathrm{e}^{I^+_{Plane}\varphi^+} + \mu \mathrm{e}^{I^-_{Plane}\varphi^-} \quad (5),$$

where

$$\varphi^\pm = \cos^{-1}\left(\frac{1}{\sqrt{2}}\cos\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)\right),$$

$$I^\pm_{Plane} = I_S \frac{\sin\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}{\sqrt{1 + \sin^2\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}} + I_{B_0} \frac{\cos\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}{\sqrt{1 + \sin^2\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}} + I_{E_0} \frac{\sin\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}{\sqrt{1 + \sin^2\omega\left(t \mp \left[(I_3 I_S)\cdot r\right]\right)}}$$

The triple of unit value basis orthonormal bivectors $\left\{I_S, I_{B_0}, I_{E_0}\right\}$ is comprised of the I_S bivector, dual to the propagation direction vector; I_{B_0} is dual to initial vector of magnetic field; I_{E_0} is dual to initial vector of electric field.

[1]Remember, in the current theory scalars are real ones. "Complex" scalars have no sense.

The expression (5) is linear combination of two geometric algebra states, g-qubits.

Linear combination of the two equally weighted basic solutions of the Maxwell equation F_+ and F_-, $\lambda F_+ + \mu F_-$ with $\lambda = \mu = 1$ reads:

$$\lambda F_+ + \mu F_-\big|_{\lambda=\mu=1} = 2\cos\omega\left[\left(I_3 I_S\right)\cdot r\right]\left(\frac{1}{\sqrt{2}}\cos\omega t + I_S\frac{1}{\sqrt{2}}\sin\omega t + I_{B_0}\frac{1}{\sqrt{2}}\cos\omega t + I_{E_0}\frac{1}{\sqrt{2}}\sin\omega t\right) \quad (6)$$

where $\cos\varphi = \frac{1}{\sqrt{2}}\cos\omega t$ and $\sin\varphi = \frac{1}{\sqrt{2}}\sqrt{1+\left(\sin\omega t\right)^2}$. It can be written in standard exponential form $\cos\varphi + \sin\varphi I_B = \mathrm{e}^{I_B\varphi}$.[2]

I will call such g-qubits *spreons* (or sprefields) because they spread over the whole three-dimensional space for all values of time and, particularly, instantly change under Clifford translations over the whole three-dimensional space for all values of time, along with the results of measurement of any observable.

3. CUDA GPU Simulation of Analog Modeling Computer

Measurement is by definition the result of action of operator, namely state, wave function, in the form of g-qubit $\left(\alpha + I_S\beta\right)$ [4], on an observable C:

$$\left(\alpha - I_S\beta\right)C\left(\alpha + I_S\beta\right) = \widetilde{\left(\alpha + I_S\beta\right)}C\left(\alpha + I_S\beta\right)$$

Take as first example the case of observable as a vector expanded in $\left\{I_3 I_S, I_3 I_{B_0}, I_3 I_{E_0}\right\} \equiv \left\{e_1, e_2, e_3\right\}$, basis vectors dual to bivectors $\left\{I_S, I_{B_0}, I_{E_0}\right\}$:

$$C = c_1 e_1 + c_2 e_2 + c_3 e_3$$

Measure it with wave function (6):

$$4\cos^2\omega\left[\left(I_3 I_S\right)\cdot r\right]\left[\left(\frac{1}{\sqrt{2}}\cos\omega t - I_S\frac{1}{\sqrt{2}}\sin\omega t - I_{B_0}\frac{1}{\sqrt{2}}\cos\omega t - I_{E_0}\frac{1}{\sqrt{2}}\sin\omega t\right)\right]$$
$$\times C\left[\left(\frac{1}{\sqrt{2}}\cos\omega t + I_S\frac{1}{\sqrt{2}}\sin\omega t + I_{B_0}\frac{1}{\sqrt{2}}\cos\omega t + \frac{1}{\sqrt{2}}I_{E_0}\sin\omega t\right)\right]$$
$$= 2\cos^2\omega\left[\left(I_3 I_S\right)\cdot r\right]\left[\left(\cos\omega t - I_S\sin\omega t - I_{B_0}\cos\omega t - I_{E_0}\sin\omega t\right)\right]$$
$$\times\left(c_1 I_3 I_S + c_2 I_3 I_{B_0} + c_3 I_3 I_{E_0}\right)\left[\left(\cos\omega t + I_S\sin\omega t + I_{B_0}\cos\omega t + I_{E_0}\sin\omega t\right)\right]$$

The result is:

$$4\cos^2\omega\left[\left(I_3 I_S\right)\cdot r\right]\left[c_3 e_1 + \left(c_1\sin 2\omega t + c_2\cos 2\omega t\right)e_2 + \left(c_2\sin 2\omega t - c_1\cos 2\omega t\right)e_3\right]$$

Geometrically that means that the measured vector is rotated by $\frac{\pi}{2}$ in the I_{B_0} plane, such that the $c_3 e_3$ component becomes orthogonal to plane I_S and remains unchanged. Two other vector components became orthogonal to I_{B_0} and I_{E_0} and continue rotating in I_S with angular velocity $2\omega t$.

[2]Good to remember that the two basic solutions F_+ and F_- differ only by the sign of $I_3 I_S$, which is caused by orientation of I_S that in its turn defines if the triple $\left\{\hat{E}, \hat{H}, \pm I_3 I_S\right\}$ is right-hand screw or left-hand screw oriented.

One another example of measurement is that of the g-qubit type observable $C_0 + C_1B_1 + C_2B_2 + C_3B_3$ (actually Hopf fibration) by a state $\alpha + \beta_1B_1 + \beta_2B_2 + \beta_3B_3$ [5]:

$$C_0 + C_1B_1 + C_2B_2 + C_3B_3 \xrightarrow{\alpha+\beta_1B_1+\beta_2B_2+\beta_3B_3} C_0$$
$$+\left(C_1\left[\left(\alpha^2+\beta_1^2\right)-\left(\beta_2^2+\beta_3^2\right)\right]+2C_2\left(\beta_1\beta_2-\alpha\beta_3\right)+2C_3\left(\alpha\beta_2+\beta_1\beta_3\right)\right)B_1$$
$$+\left(2C_1\left(\alpha\beta_3+\beta_1\beta_2\right)+C_2\left[\left(\alpha^2+\beta_2^2\right)-\left(\beta_1^2+\beta_3^2\right)\right]+2C_3\left(\beta_2\beta_3-\alpha\beta_1\right)\right)B_2$$
$$+\left(2C_1\left(\beta_1\beta_3-\alpha\beta_2\right)+2C_2\left(\alpha\beta_1+\beta_2\beta_3\right)+C_3\left[\left(\alpha^2+\beta_3^2\right)-\left(\beta_1^2+\beta_2^2\right)\right]\right)B_3$$

with:

$$B_1 = I_S,\ B_2 = I_{B_0},\ B_3 = I_{E_0},\ \alpha = 2\cos\omega\left[\left(I_3I_S\right)\cdot r\right]\frac{1}{\sqrt{2}}\frac{1}{\sqrt{2}}\cos\omega t,$$
$$\beta_1 = 2\cos\omega\left[\left(I_3I_S\right)\cdot r\right]\frac{1}{\sqrt{2}}\sin\omega t,\ \beta_2 = 2\cos\omega\left[\left(I_3I_S\right)\cdot r\right]\frac{1}{\sqrt{2}}\cos\omega t,$$
$$\beta_3 = 2\cos\omega\left[\left(I_3I_S\right)\cdot r\right]\frac{1}{\sqrt{2}}\sin\omega t$$

gives a G_3^+ element spreading through the three-dimensional space for all values of time parameter t:

$$4\cos^2\omega\left[\left(I_3I_S\right)\cdot r\right]\left[C_0 + C_3I_S + \left(C_1\sin 2\omega t + C_3\cos 2\omega t\right)I_{B_0}\right.$$
$$\left.+\left(C_2\sin 2\omega t - C_1\cos 2\omega t\right)I_{E_0}\right]$$

The current approach transcends common quantum computing schemes since the latter have tough problems of creating large sets of (entangled) qubits. In the current scheme any test observable can be placed into continuum of the (t,r) dependent values of the spreon state. All other observables measurement results are connected to a measured observable by Clifford translations thus giving any amount of the observables values spread over three dimensions and at all instants of time not generally following cause/effect ordering.

The sprefield hardware requires special implementation as a photonic/laser device. Instead, an equivalent simulation scheme can be used where the amount of available space points there is only restricted by the overall available Nvidia GPU number of threads.

In the case of measuring a vector the three parallel calculated measured by the sprefield vector components $4\cos^2\omega\left[\left(I_3I_S\right)\cdot r\right]c_3$, $4\cos^2\omega\left[\left(I_3I_S\right)\cdot r\right]\left(c_1\sin 2\omega t + c_2\cos 2\omega t\right)$ and $4\cos^2\omega\left[\left(I_3I_S\right)\cdot r\right]\left(c_2\sin 2\omega t - c_1\cos 2\omega t\right)$ are processed using the following fragmental variants of CUDA code:

```
uint width = 512, height = 512;//optional value
dim3 blockSize(16, 16);//optional value

__global__ void quantKernel(float3* output, int dimx, int dimy, int dimz,
float t)
{
```

```
float c1 = 1.0; //components of observable vectors
float c2 = 1.0;
float c3 = 1.0;
float omega = 12560000.0;// possible angular velocity in the sprefield

float tstep = 1.0f;

float factor = 0.0;

int qidx = threadIdx.x + blockIdx.x * blockDim.x;
int qidy = threadIdx.y + blockIdx.y * blockDim.y;
int qidz = threadIdx.z + blockIdx.z * blockDim.z;

size_t oidx = qidx + qidy*dimx + qidz*dimx*dimy;

output[oidx][0] = oidx*tstep;
factor =4*(cosf(omega * output[oidx][0])) * (cosf(omega * output[oidx][0]));
output[oidx][0] += factor * c3;
output[oidx][1] = oidx*tstep;
output[oidx][1] += factor * (c1 sin(2 * omega * t) + c2 cos (2 * omega * t));
output[oidx][2] = oidx*tstep;
output[oidx][2] += factor * (c2 sin(2 * omega * t) + c1 cos (2 * omega * t));
}

template<typename T>
void init(char * devPtr, size_t pitch, int width, int height, int depth)
{
size_t qPitch = pitch * height;
int v = 0;
for (int z = 0; z < depth; ++z) {
char * slice = devPtr + z * qPitch;;
for (int y = 0; y < height; ++y) {
        T * row = (T *)(slice + y * pitch);
for (int x = 0; x < width; ++x) {
            row[x] = T(v++);
            }
        }
    }

int keyboard(unsigned char key)
{
switch (key)
    {
case (27):
cudaFree(d_output);
```

```
                    free(h_output);
  return 1;
                    break;
  default:
  break;

    }
  }

  int main(void)
  {
    VolumeType *h_volumeMem;
  unsigned char key;
  __device__ float g_fAnim = 0.0;
  float3* h_output = (float3*)malloc(size * sizeof(float3));//array for measured vector //observable values
  float3* d_output = NULL;

  checkCudaErrors(
    cudaMalloc((void **)&d_output, width * height * sizeof(float3)));
  checkCudaErrors(cudaMemset(d_output, 0, width * height * sizeof(float3)));

    cudaExtent volumeSizeBytes = make_cudaExtent(width, SIZE_Y, SIZE_Z);
    cudaPitchedPtr d_volumeMem;
  checkCudaErrors (cudaMalloc3D(&d_volumeMem, volumeSizeBytes));

  size_t size = d_volumeMem.pitch * SIZE_Y * SIZE_Z;
    h_volumeMem = (VolumeType *)malloc(size);
  init<VolumeType>((char *)h_volumeMem, d_volumeMem.pitch, SIZE_X, SIZE_Y, SIZE_Z);
  checkCudaErrors (cudaMemcpy(d_volumeMem.ptr, h_volumeMem, size, cudaMemcpyHostToDevice));

    cudaArray * d_volumeArray;
    cudaChannelFormatDesc channelDesc = cudaCreateChannel-Desc<VolumeType>();
    cudaExtent volumeSize = make_cudaExtent(SIZE_X, SIZE_Y, SIZE_Z);
  checkCudaErrors ( cudaMalloc3DArray(&d_volumeArray, &channelDesc, volumeSize) );

    cudaMemcpy3DParms copyParams = {0};
    copyParams.srcPtr = d_volumeMem;
    copyParams.dstArray = d_volumeArray;
    copyParams.extent = volumeSize;
    copyParams.kind = cudaMemcpyDeviceToDevice;
```

```
checkCudaErrors ( cudaMemcpy3D(&copyParams) );
   while(1)
   {
g_fAnim += 0.01f;
quantKernel<<<1,dim3(4,4,4)>>>(d_output,4,4,4, g_fAnim);
cudaError_t error = cudaGetLastError();

checkCudaErrors (cudaMemcpy(h_output, d_output, osize, cudaMemcpyDeviceToHost));
if (keyboard(key)==1)
                    exit(error);
   }

return error;
}
```

The GPU CUDA case of simulation of measurement of a g-qubit type of observable differs in quantKernel GPU executed function:

```
__global__ void quantKernel(float4* output, int dimx, int dimy, int dimz, float t)
{

float c1 = 1.0; //components of observable vectors
float c2 = 1.0;
float c3 = 1.0;
float omega = 12560000.0;// variant of angular velocity in the sprefield

float tstep = 1.0f;

float factor = 0.0;

   int qidx = threadIdx.x + blockIdx.x * blockDim.x;
int qidy = threadIdx.y + blockIdx.y * blockDim.y;
int qidz = threadIdx.z + blockIdx.z * blockDim.z;

size_t oidx = qidx + qidy*dimx + qidz*dimx*dimy;

output[oidx][0] = oidx*tstep;
factor =4*(cosf(omega * output[oidx][0])) * (cosf(omega * output[oidx][0]));
output[oidx][0] += factor * c3;
output[oidx][1] = oidx*tstep;
output[oidx][1] += factor * (c1sin(2*omega*t)+c2cos(2*omega*t));
output[oidx][2] = oidx*tstep;
output[oidx][2] += factor * (c2sin(2*omega*t)-c1cos(2*omega*t));
output[oidx][3] = factor;
}
```

More flexibility in measurements can be achieved by scattering of the sprefield wave function before applying to observables.

Arbitrary Clifford translation $\mathrm{e}^{I_{B_C}\gamma} = \cos\gamma + \sin\gamma\left(\gamma_1 I_S + \gamma_2 I_{B_0} + \gamma_3 I_{E_0}\right)$ acting on spreons (6) gives:

$$\begin{aligned} & 2\cos\omega\left[\left(I_3 I_S\right)\cdot r\right]\left[\frac{1}{\sqrt{2}}\left(\cos\gamma\cos\omega t - \gamma_1\sin\gamma\sin\omega t - \gamma_2\sin\gamma\cos\omega t - \gamma_3\sin\gamma\sin\omega t\right)\right. \\ & +\frac{1}{\sqrt{2}}\left(\cos\gamma\sin\omega t + \gamma_1\sin\gamma\cos\omega t - \gamma_2\sin\gamma\sin\omega t + \gamma_3\sin\gamma\cos\omega t\right) I_S \\ & +\frac{1}{\sqrt{2}}\left(\cos\gamma\cos\omega t + \gamma_1\sin\gamma\sin\omega t + \gamma_2\sin\gamma\cos\omega t - \gamma_3\sin\gamma\sin\omega t\right) I_{B_0} \\ & \left.+\frac{1}{\sqrt{2}}\left(\cos\gamma\sin\omega t - \gamma_1\sin\gamma\cos\omega t + \gamma_2\sin\gamma\sin\omega t + \gamma_3\sin\gamma\cos\omega t\right) I_{E_0}\right] \end{aligned} \tag{7}$$

This result is defined for all values of t and r, in other words the result of Clifford translation instantly spreads through the whole three-dimensions for all values of time.

The instant of time when the Clifford translation was applied makes no difference for the state (7) because it is simultaneously redefined for all values of t. The values of measurements $O\left(C_0, C_1, C_2, C_3, I_S, I_{B_0}, I_{E_0}, \gamma, \gamma_1, \gamma_2, \gamma_3, \omega, t, r\right)$ also get instantly changed for all values of time of measurement, even if the Clifford translation was applied later than the measurement. That is an obvious demonstration that the suggested theory allows indefinite event casual order. In that way the very notion of the concept of cause and effect, ordered by time value increasing, disappears.

Since general result of measurement when Clifford translation takes place in an arbitrary plane is pretty complicated, I am only giving the result for the special case $\gamma_1 = 1$ and $\gamma_2 = \gamma_3 = 0$ (Clifford translation acts in plane I_S). The result is:

$$\begin{aligned} & O\left(C_0, C_1, C_2, C_3, I_S, I_{B_0}, I_{E_0}, \gamma, \gamma_1, \gamma_2, \gamma_3, \omega, t, r\right)_{\gamma_1=1, \gamma_2=\gamma_3=0} \\ & = 4\cos^2\omega\left[\left(I_3 I_S\right)\cdot r\right]\left[C_0 + \left(C_2\sin 2\gamma + C_3\cos 2\gamma\right) I_S\right. \\ & \quad + \left(C_1\sin 2\omega t + \sin 2\gamma\cos 2\omega t\left(C_2 + C_3\right)\right) I_{B_0} \\ & \quad \left. + \left(-C_1\cos 2\omega t + \sin 2\gamma\sin 2\omega t\left(C_2 - C_3\right)\right) I_{E_0}\right] \end{aligned}$$

The only component of measurement, namely the one lying in plane I_S, does not change with time. The I_{B_0} and I_{E_0} components do depend on the time of measurement being modified forward and backward in time if Clifford translation is applied. Clifford translation modifies measurement results of the past and the future.

4. Conclusion

In the suggested theory all measured observable values get available all together, not through looking one by one. In this way quantum computer appeared to be a kind of analog computer keeping and instantly processing information by and

on sets of objects possessing an infinite number of degrees of freedom. The multithread GPUs bearing the CUDA language functionality allow to simultaneously calculate observable measurement values at a number of space/time discrete points, forward and backward in time, the number only restricted by the GPU threads capacity. That eliminates the tough hardware problem of creating huge and stable arrays of qubits, the base of quantum computing in conventional approaches.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Ulmann, B. (2022) Analog Computing. De Gruyter, Oldenbourg. https://doi.org/10.1515/9783110787740

[2] Soiguine, A. (2019) Instantaneous Spreading of the g-Qubit Fields. *Journal of Applied Mathematics and Physics*, **7**, 591-602. https://doi.org/10.4236/jamp.2019.73043

[3] Soiguine, A. (2020) Scattering of Geometric Algebra Wave Functions and Collapse in Measurements. *Journal of Applied Mathematics and Physics*, **8**, 1838-1844. https://doi.org/10.4236/jamp.2020.89138

[4] Soiguine, A. (2020) The Geometric Algebra Lift of Qubits and Beyond. LAMBERT Academic Publishing, Saarbrucken.

[5] Soiguine, A. (2015) Geometric Phase in Geometric Algebra Qubit Formalism. Lambert Academic Publishing, Saarbrucken.