МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ

Розрахунково графічна робота

По предмету: “Програмне забезпечення спеціалізованих систем”

Реалізація ускладнених математичних

функцій (arcsin(0,7\*x))

Виконав:

студент групи ЗАМ-171 Борщов М. I.

Перевірив: Ступень П. В

Одеса 2022

# **Введення**

Метою розрахунково-графічної роботи є розвиток індивідуальних практичних навиків проектування програмного забезпечення для спеціалізованих комп’ютерних систем на основі цифрових процесорів обробки сигналів.

# **Аналіз поставленого завдання**

Відповідно до варіанту №3 необхідно розробити програму, яка буде розраховувати вхідне число “Х” відповідно до функції arcsin(0,7\*x). Тип процесору ADSP 2106X (SHARC)

Вихідними даними буде результат розрахунків.

Функція Арксинусом числа а називають сема число з проміжку [−π2;π2], синус якого дорівнює цьому числу.

Розглянемо, як програмно апроксимувати функцію арксинусу.

Arcsin(0,7\*x) обчислюється в чотири етапи:

1. Аргумент x (який може бути будь-яким дійсним значенням) помножується на 0.7, при чому перевіряється діапазон - t має діапазон ±2π.

2. Arcsin(f) обчислюється за допомогою функції:

Src

asin(Src) = atan ---------------- -> Dest

sqrt(1-Src^2)

3. Tan(x) обчислюється в три етапи:

1. Аргумент x (який може бути будь-яким дійсним значенням) зводиться до відповідного аргументу f із величиною, меншою за π/4 (тобто t має діапазон ± π/2).

2. Tan( f) обчислюється за допомогою поліноміальної апроксимації min-max.

3. Функія реалізується

4. функція котангенсу обчислюється прри зміні поліному:

cotan ( f) = Q(g)/–f \* P(g)

Спочатку аргумент x зводиться до аргументу f. Це зменшення аргументу виконується в розділах, позначених compute modulo та compute f. Коефіцієнт π/2 потрібен під час обчислення, щоб зменшити x (яке може бути будь-яке значення з плаваючою точкою) до f (яке є нормованим значенням із діапазоном ± π/2). Щоб отримати точний результат, константи C1 і C2 вибираються так, щоб C1 + C2 наближалися до π/2 з трьома або чотирма знаками після коми за межами машинної точності. Значення C1 обрано таким чином, щоб бути близьким до π/2, а C2 – це коефіцієнт, який додається до C1, що призводить до дуже точного представлення π/2.

# **Проєктування програми**

#include "asm\_glob.h"

.SEGMENT/PM Assembly\_Library\_Code\_Space;

.PRECISION=MACHINE\_PRECISION;

.GLOBAL tan;

tan: i\_reg=tangent\_data;

F8=PASS F0, F2=mem(i\_reg,1); { Use absolute value of input }

compute\_modulo: F4=F8\*F2, F1=F0; { Compute fp modulo value }

R2=FIX F4, F12=mem(i\_reg,1); { Rnd nearest fractional portion }

F4=FLOAT R2, R0=R2; { Return to fp }

compute\_f: F12=F12\*F4, F2=mem(i\_reg,1); { Compute XN\*C1 }

F2=F2\*F4, F12=F8-F12; { Compute X-XN\*C1, and XN\*C2 } F8=F12-F2, F4=mem(i\_reg,1); { Compute f=(X-XN\*C1)-XN\*C2 }

F12=ABS F8, F7=F8;

F4=F12-F4, F12=mem(i\_reg,1); { Check for TAN(x)=x }

IF LT JUMP compute\_quot; { Compute quotient with NUM=f DEN=1 }

compute\_P: F12=F8\*F8, F4=mem(i\_reg,1); { g=f\*f }

F4=F12\*F4, F2=mem(i\_reg,1); { Compute p3\*g }

F4=F2+F4; { Compute (p3\*g + p2) }

F4=F12\*F4, F2=mem(i\_reg,1); { Compute (p3\*g + p2)\*g }

F4=F2+F4; { Compute (p3\*g + p2)\*g + p1 }

F4=F12\*F4; { Compute ((p3\*g + p2)\*g + p1)\*g }

F4=F4\*F8; { Compute ((p3\*g + p2)\*g + p1)\*g\*f }

F8=F4+F8, F4=mem(i\_reg,1); { Compute f\*P(g) }

compute\_Q: F4=F12\*F4, F2=mem(i\_reg,1); { Compute sum\*g }

F4=F2+F4; { Compute sum=sum+next q }

F4=F12\*F4, F2=mem(i\_reg,1); { Compute sum\*g }

F4=F2+F4; { Compute sum=sum+next q }

F4=F12\*F4, F2=mem(i\_reg,1); { Compute sum\*g }

F12=F2+F4, F7=F8; { Compute sum=sum+next q }

compute\_quot:BTST R0 BY 0;

IF NOT SZ F12=-F7, F7=F12;

F0=RECIPS F12; { Get 4 bit seed R0=1/D }

F12=F0\*F12, F11=mem(i\_reg,1); { D(prime) = D\*R0

F7=F0\*F7, F0=F11-F12; { F0=R1=2-D(prime),

F7=N\*R0 } F12=F0\*F12; { F12=D(prime)=D(prime)\*R1 }

F7=F0\*F7, F0=F11-F12; { F7=N\*R0\*R1, F0=R2=2-D(prime) }

RTS (DB), F12=F0\*F12; { F12=D(prime)=D(prime)\*R2 }

F7=F0\*F7, F0=F11-F12; { F7=N\*R0\*R1\*R2, F0=R3=2-D(prime) }

F0=F0\*F7; { F7=N\*R0\*R1\*R2\*R3 }

compute\_value\_for\_tan: {0.7 \* (Src / sqrt(1-Src^2))}

F2=mem(i\_reg,1); { get input\_value }

F3=F2\*F2; { Calculate square value }

F0=Q0 - F2; { 1 - Src^2 }

F1=sqrt(F0); { Calculate square value }

F5=F2 / F1; {divide value }

F7=mem(i\_reg,14)\*F5; { Calculate input value }

.ENDSEG;

.SEGMENT/SPACE Assembly\_Library\_Data\_Space; .PRECISION=MEMORY\_PRECISION;

.VAR tangent\_data[14] = 0.6366197723675834308, { 2/PI }

1.57080078125, { C1, almost PI/2 }

-4.454455103380768678308E-6, { C2, PI/2=C1+C2 }

9.536743164E-7, { eps, TAN(eps)=eps }

1.0, { Used in one path }

-0.7483634966612065149E-5, { P3 }

0.2805918241169988906E-2, { P2 }

-0.1282834704095743847, { P1 }

-0.2084480442203870948E-3, { Q3 }

0.2334485282206872802E-1, { Q2 }

-0.4616168037429048840, { Q1 }

1.0, { Q0 }

2.0; { Used in divide }

0.7; { coefficient of initial data }

.ENDSEG;

# **Висновок**

Для розрахунку арксинусу значення необхідно провести розрахунки у декілька єтапів. При розрахунках використовується функція котангенсу. Программа виконує необхідний функціонал. Треба бути уважним до крайніх значень та перевіряти діапазон даних, які подаються на вхід.

При розрахунках виникає невелика похибка, так як при розрахунках певні значення округлюються.