



-- *Lluís Parcerisa* --

C++ as a programming language for CAS

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions

- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions

- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

C++ vs Basic - Features

Basic

C++

High Level

Common

Versatility

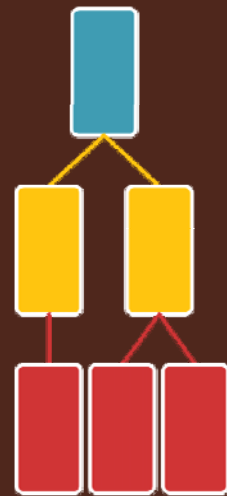
Complexity

High Level

Efficient

C++ vs Basic – App. Structure

- C++ => Object oriented
 - Code reuse and extension
 - Adequate for complex systems
 - Empower visual applications
 - Prototype construction
 - Speed application programming time
 - Enforce cooperative work
 - Software maintenance



C++ to program calculator apps



Programmer

- Fast learning
- Code division
- Embedded functions
- CAS calling



User

- Easy
- Intuitive
- Useful
- Customizable

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

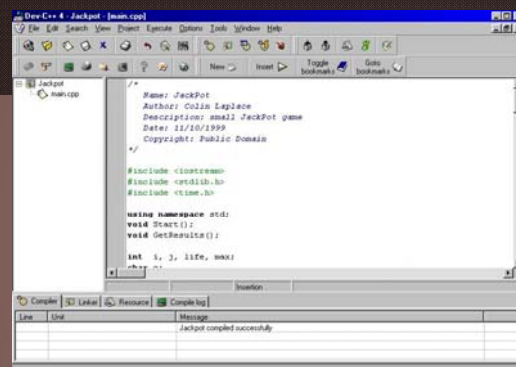
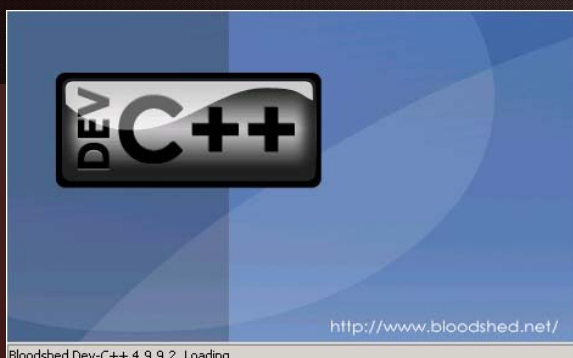
- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions

- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

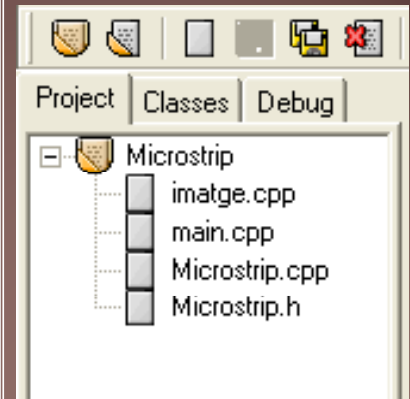
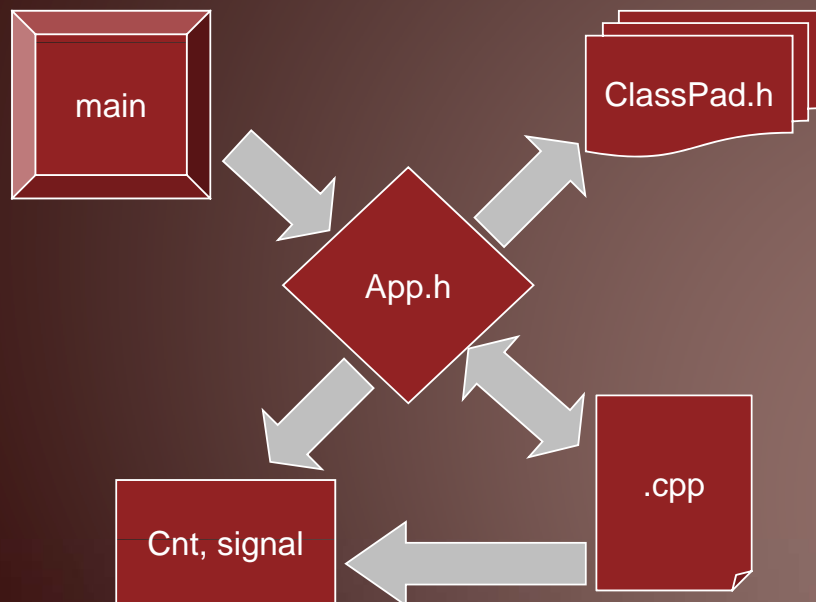
Development Environment

- C Syntax
- Compilation: add-in cpa
- Software: Dev-CPP
- www.cpsdk.com



Application Structure

- Model: A. Pizdrowsky / B. White



Implementation example

```
main.cpp
#include "Microstrip.h"
// The main method
void PegAppInitialize(PegPresentationManager *pPresentation)
{
    Microstrip *swin; //create the program as an object
    PegRect Rect; // create a rectangle object for the MainFrame
    Rect.Set(MAINFRAME_LEFT, MAINFRAME_TOP, MAINFRAME_RIGHT,
    MAINFRAME_BOTTOM); // set the rectangle to the usable size of the display
    CPMainFrame *mw = new CPMainFrame(Rect); // create the MainFrame
    PegRect AppRect; // create a rectangle for the Application Window
    AppRect = mw->FullAppRectangle(); // size it to the MainFrame
    swin = new Microstrip(AppRect,mw); // create the program as an Application Window
    mw -> SetTopWindow(swin); // load the application window into the MainFrame
    mw -> SetMainWindow(swin); // set a main window for this module
    pPresentation->Add(mw); // add the MainFrame to the Presentation Manager
}
```

Implementation example

```
Microstrip.h
#include "ClassPad.h"
#define BOTO 10

class Microstrip: public CPMModuleWindow
{
protected:
CPegString* T;
CPegString* W;
CPegString* H;
CPegString* er;
PegPrompt* res1;
PegPrompt* res2;
PegPrompt* res3;
PegPrompt* res4;
OBCD Zo;
OBCD Co;
OBCD Lo;
OBCD Tpd;

public: Microstrip(PegRect, CPMMainFrame*);
~Microstrip(){}
int init();
void Draw();
SIGNED Message(const PegMessage &Mesg);
WORD Dialog(char*, char*);
void Calcula();
};
```

Implementation example

```
Microstrip.cpp
#include "Microstrip.h"

extern PegBitmap gbmatgeBitmap;

Microstrip::Microstrip(PegRect rect, CPMMainFrame* frame):CPModuleWindow(rect,0,0,frame)
{
init();
}

int Microstrip::init()
{
const WORD wStyle = FF_NONE|TJ_LEFT|AF_TRANSPARENT| TT_COPY;
const SIGNED margin = 5;
const SIGNED width=mClient.Width()-2*margin;

PEGCHAR* chrepsilon = "\xEC\xA0";
PEGCHAR* chrmu = "\xED\x72";

PegRect rectangle;

PegPrompt* label5 = new PegPrompt(margin, 80, width/2, "W("+CPString(chrmu)+"m):",0,wStyle);
W = new CPegString(margin+41,80,30,"",0, FF_THIN|AF_ENABLED|EF_EDIT);
Addr(W);
Addr(label5);

PegPrompt* label = new PegPrompt(margin, 95, width/2, "T("+CPString(chrmu)+"m):",0,wStyle);
T = new CPegString(margin+41,95,30,"",0, FF_THIN|AF_ENABLED|EF_EDIT);
Addr(T);
Addr(label);
}
```


Implementation example

Microstrip 300 Fri Oct 12 19:07:08 2007

File Edit View

Diagram of a microstrip structure with dimensions L , T , H , W , and ϵ_r .

$W(\mu m) = 3$ $H(\mu m) = 4$
 $T(\mu m) = 1$ $\epsilon_r = 2$

Calcular!

$Z_0 = 91.91 \Omega$
 $C_0 = 0.461 \text{ pF/cm}$
 $L_0 = 3.895 \text{ nH/cm}$
 $T_{pd} = 4.243 \text{ ns/m}$

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

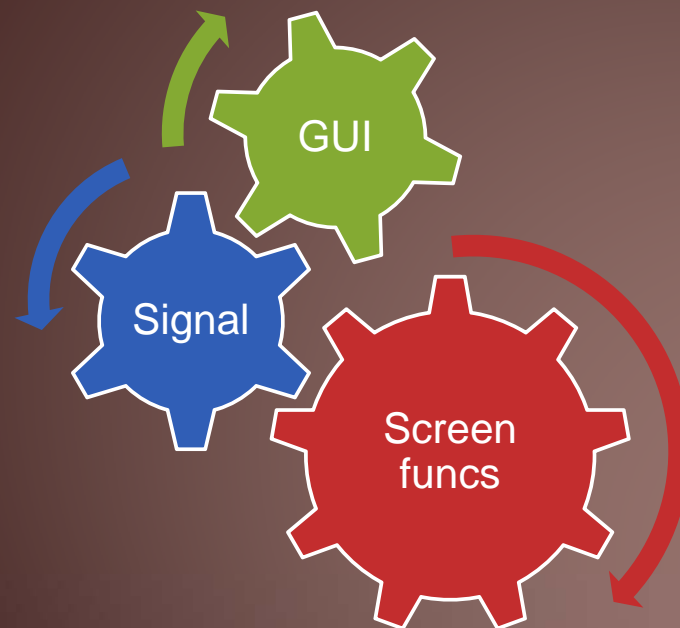
4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions

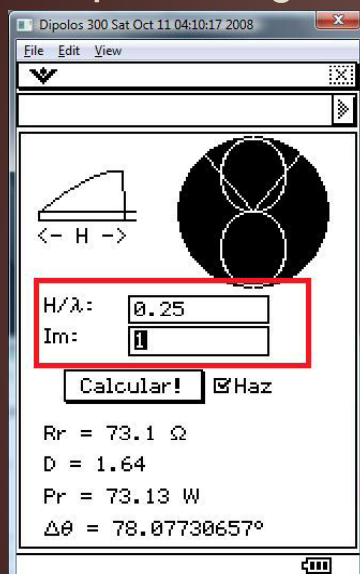
- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

Embedded Functions

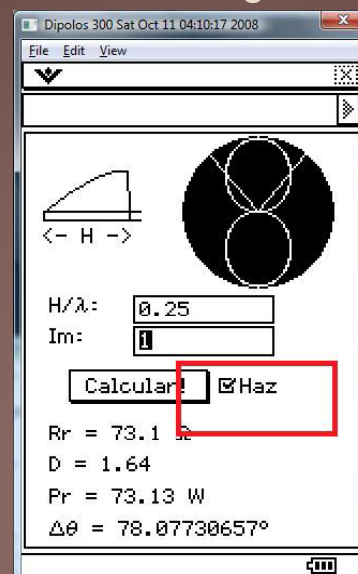


GUI Elements

PegPrompt, CPPegString

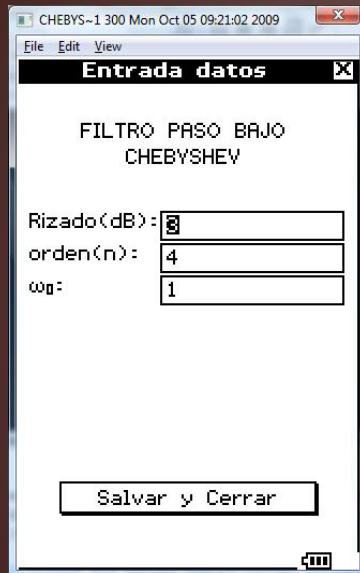


PegCheckBox, PegTextButton

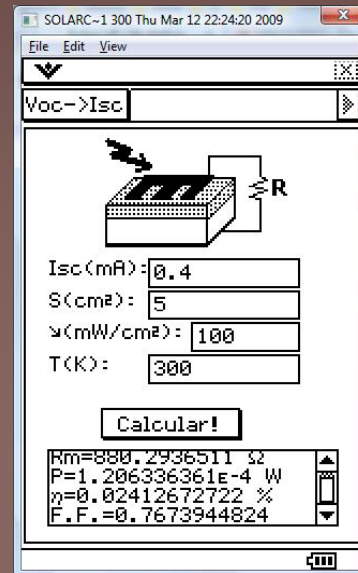


GUI Elements

CPDialog, PegMessageWindow



PegVertList, PegComboBox



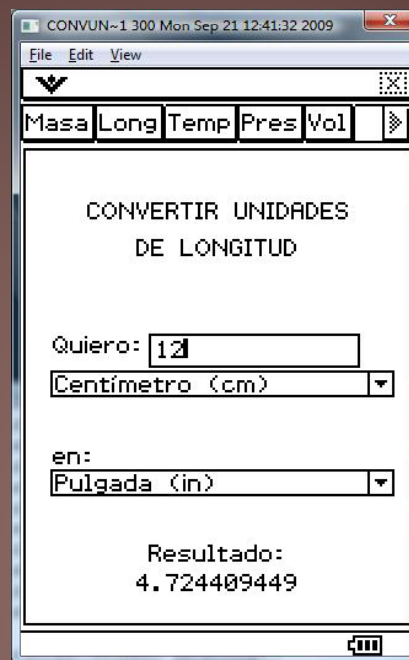
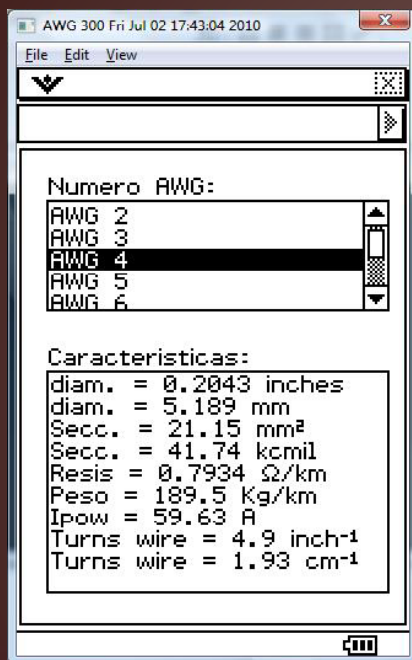
Signals

```
pList->SetSignals(SIGMASK(PSF_LIST_SELECT));
```

```
(...)
```

```
SIGNED AWG::Message(const PegMessage &Mesg) {
    switch(Mesg.wType) {
        case SIGNAL(ID_VERTLIST,PSF_LIST_SELECT): {
            cList->Clear();
            Calcula(pList->GetSelectedIndex()-3);
            Draw();
        } break;
        default: return CPMModuleWindow::Message(Mesg);
    }
    return 0;
}
```

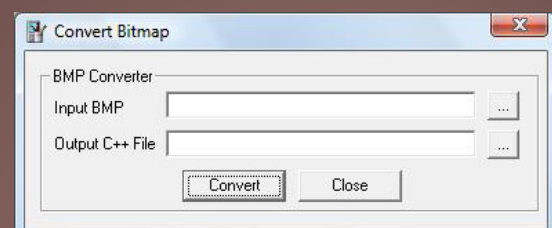
Ex: AWG, Conversor



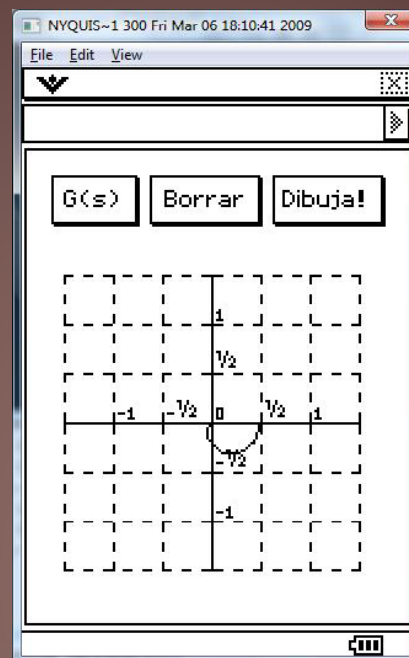
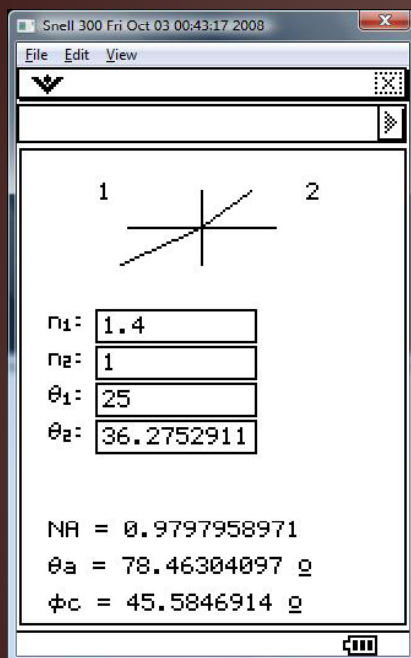
Graphical Functions

- PegScreen
 - Line
 - Rectangle
 - Circle
 - PatternLine
 - Arc
 - Polygon

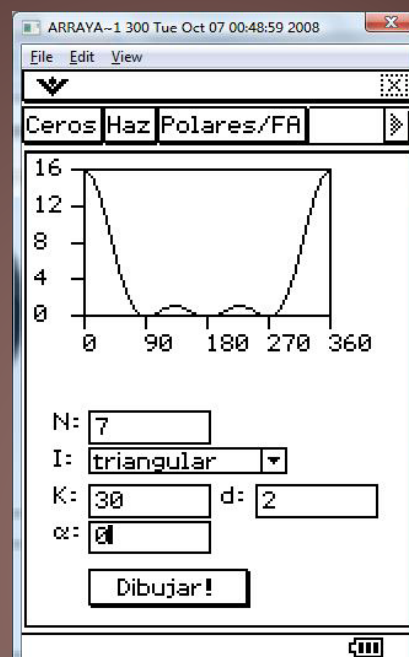
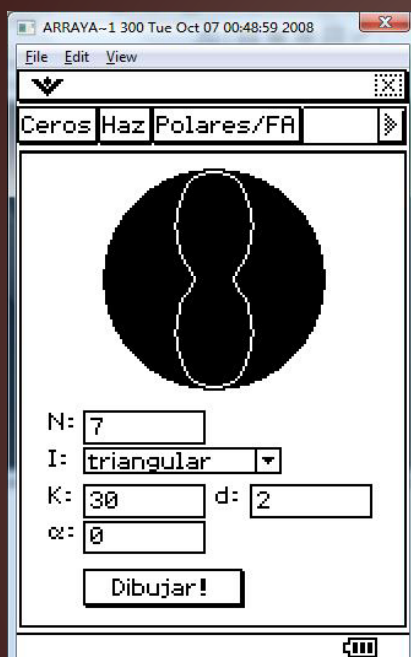
- Convert Bitmap



Ex1: Snell, Nyquist Diagram



Ex2: Antenna Arrays



Real numbers

OBCD Data Structure

The structure of an OBCD is defined as:

```
typedef struct obcd_ {  
    unsigned char mantissa[IM_CAL_INDIGIT];  
    unsigned short exponential;  
} OBCD_;  
  
typedef union obcd {  
    OBCD_ obcd1;  
    unsigned long    dummy[3];  
} OBCD;
```

The mantissa of a number is stored in obcd1.mantissa. The mantissa is 10 bytes long, with the least significant 2 bytes reserved for system use. The most significant nibble is reserved for a flag. There is also a 2 byte exponent that is stored in a short. The entire structure looks like this:

eF	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	eS	e1	e2	e3
Flag	Mantissa															reserved				Exponent			

Real numbers

Functions

word Cal_defrac_OBC (OBCD *x, OBCD *y, OBCD *z)
Converts a decimal to a fraction.

word Cal_add_OBC (OBCD *x, OBCD *y)
Adds x to y ($x+y$).

word Cal_sub_OBC (OBCD *x, OBCD *y)
Subtracts x from y ($x-y$).

word Cal_mul_OBC (OBCD *x, OBCD *y)
Multiplies x and y ($x*y$).

word Cal_div_OBC (OBCD *x, OBCD *y)
Divides y into x (x/y).

word Cal_pow_OBC (OBCD *x, OBCD *y)
Raises x to y (x^y).

word Cal_relat (OBCD *x, OBCD *y)
Compares x to y to determine an inequality relationship.

word Cal_relat0 (OBCD *x)
Compares x to 0.

word Cal_relateq (OBCD *x, OBCD *y)
Compares x to y and returns equal or not equal.

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

5.- Conclusions

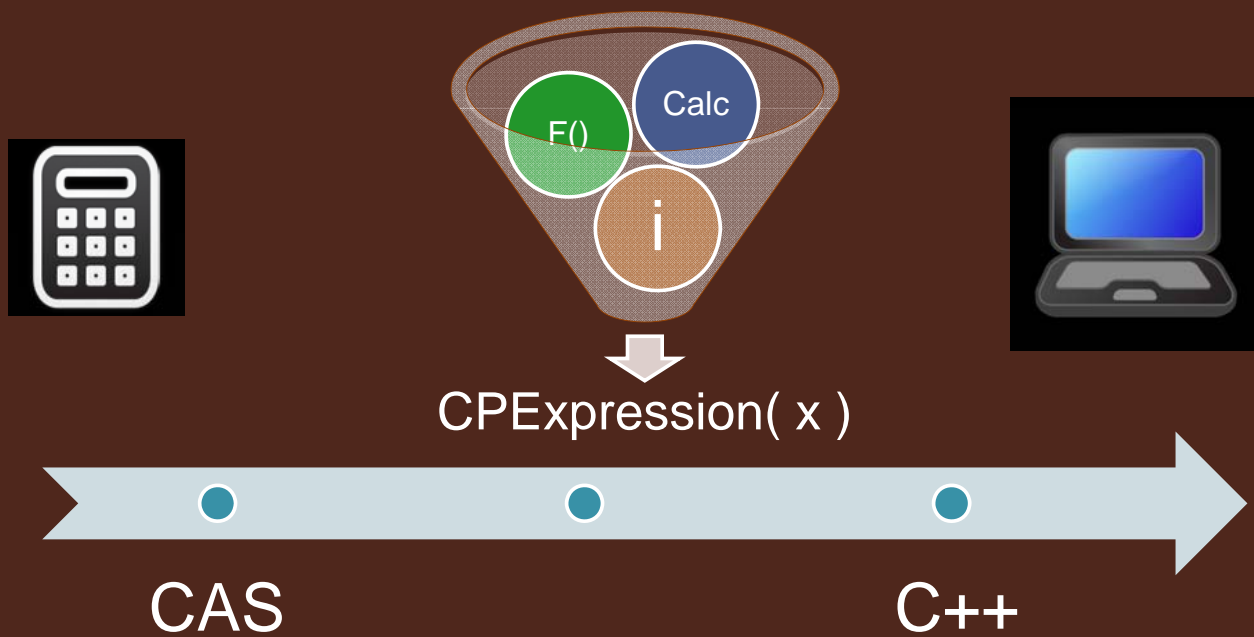
- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

Calling CAS Functions

• CPEXpression Class Reference

CPP expression class which manages expression memory. The CPEXpression class contains a reference to a CPEXpressionObject. A CPEXpressionObject should never be used directly. This is a reference counted object that keeps track of the memory used for the CAS expression. When there are no longer any CPEXpression objects referring to this memory it is automatically freed.

Calling CAS Functions



Calling CAS Functions

Transform	Fraction	Approx / toFrac / propFrac	3/2 => 1.5 1.5 => 3/2 1.5 => 1 1/2
Calculate	Polinomy	Expand / Collect Factor / factorOut rFactor Combine Simplify	(a+b)^2 => ... Factorize (x) Root mcd
Complexes			
Matrices			
Equations			
Trigonometry		tExpand / tCollect expToTrig / trigToExp	sin(a+b) e^jx
Other		Dms / toDMS	3.44 => 3,26,24

Calling CAS Functions

Transform

Calculate

Complexes

Matrices

Equations

Taylor

```
taylor(sin(x),x,5,0)
```

$$\frac{x^5}{120} - \frac{x^3}{6} + x$$

Laplace / InvLaplace

$$L[f(t)](s) = \int_0^{\infty} f(t)e^{-st} dt$$

```
laplace(x^2+2x=e^-t,t,x,s)
-x(0)+Lp.s+2.Lp=1/(s+1)
ans|x(0)=3
Lp.s+2.Lp-3=1/(s+1)
solve(ans,Lp)
{Lp=3.s/(s^2+3.s+2)+4/(s^2+3.s+2)}
invlaplace(getright(ans[1]),s,t)
e^-t+2.e^-2.t
```

+ Fourier/InvFourier/FFT...

Calling CAS Functions

Transform

Calculate

Complexes

Matrices

Equations

diff / \int

lim

Σ / Π

fMin / fMax

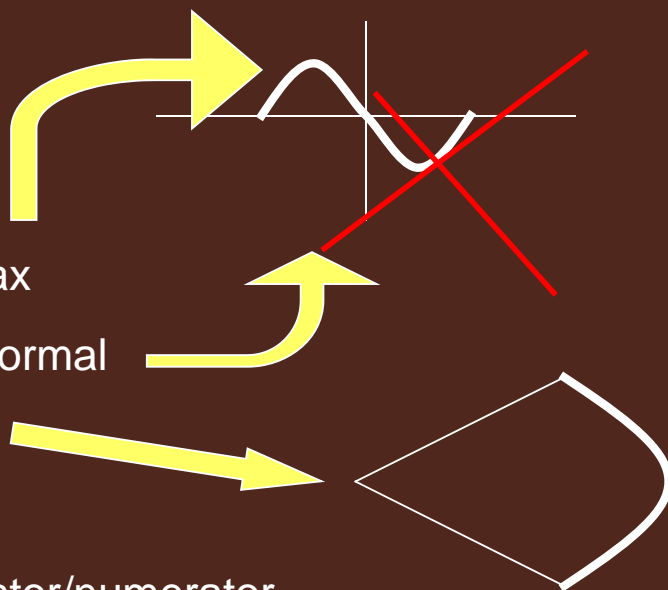
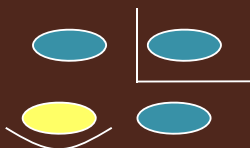
tanLine/normal

arcLine

gcd / lcm

denominator/numerator

mod



Calling CAS Functions

Transform

Calculate

Complexes

Matrices

Equations

$$\sqrt{a^2 + b^2}$$

$$\arctg(b/a)$$

$$a - jb$$

$$a / b$$

Abs

Arg

Conjg

Im / Re

$$a + jb$$

cExpand

compToTrig

compToPol

$$\sqrt{a^2 + b^2} \cdot (\cos() + j \sin())$$

compToPol⁻¹

$$\sqrt{a^2 + b^2} \cdot e^{(j \arctg(b/a))}$$

FLAMAGAS
DIVISION DIDACTICA
www.aulaCASIO.com



Calling CAS Functions

Transform

Calculate

Complexes

Matrices

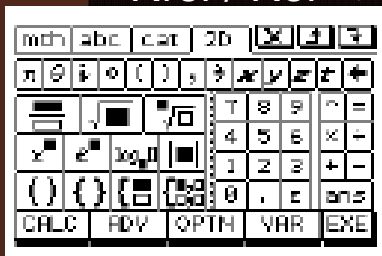
Equations

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- Trn => [a c][b d]
- Dim => {2,2}
- Det => ad - bc
- Norm => $\sqrt{|a|^2 + |b|^2 + |c|^2 + |d|^2}$
- eigVl / eigVc => propis
- Rref / Ref => escalonada

Creation:

- Menú
- Vectors
- 2D



descomposició

[a b]

FLAMAGAS
DIVISION DIDACTICA
www.aulaCASIO.com



Calling CAS Functions

Transform

Calculate

Complexes

Matrices

Equations

Reals

`Solve(3x-8=0)`

Parameter

`Solve(mx^2+x-5=0)`

Complexes

`Solve(y^2-1=0,y)`

`Solve(x-5>2)`

Inequalities

`Solve(x^2≠5)`

`Solve({x+y=0,x-y=7},{x,y})`

Systems

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

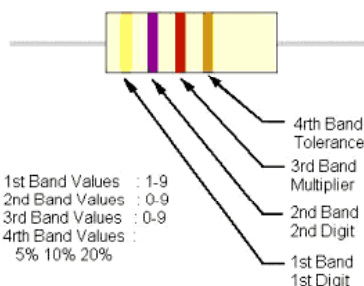
5.- Conclusions

- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

Lab - Standard

E6	1.0			1.5			2.2			3.3			4.7			6.8																																																																																																																																																																								
E12	1.0	1.2	1.5	1.8	2.2	2.7	3.3	3.9	4.7	5.6	6.8	8.2																																																																																																																																																																												
E24	1.0	1.1	1.1	1.1	1.2	1.2	1.2	1.3	1.3	1.3	1.4	1.4	1.5	1.5	1.6	1.6	1.7	1.7	1.8	1.8	1.9	1.9	2.0	2.0	2.1	2.1	2.2	2.2	2.3	2.3	2.4	2.4	2.5	2.5	2.6	2.6	2.7	2.7	2.8	2.8	2.9	2.9	3.0	3.0	3.1	3.1	3.2	3.2	3.3	3.3	3.4	3.4	3.5	3.5	3.6	3.6	3.7	3.7	3.8	3.8	3.9	3.9	4.0	4.0	4.1	4.1	4.2	4.2	4.3	4.3	4.4	4.4	4.5	4.5	4.6	4.6	4.7	4.7	4.8	4.8	4.9	4.9	5.0	5.0	5.1	5.1	5.2	5.2	5.3	5.3	5.4	5.4	5.5	5.5	5.6	5.6	5.7	5.7	5.8	5.8	5.9	5.9	6.0	6.0	6.1	6.1	6.2	6.2	6.3	6.3	6.4	6.4	6.5	6.5	6.6	6.6	6.7	6.7	6.8	6.8	6.9	6.9	7.0	7.0	7.1	7.1	7.2	7.2	7.3	7.3	7.4	7.4	7.5	7.5	7.6	7.6	7.7	7.7	7.8	7.8	7.9	7.9	8.0	8.0	8.1	8.1	8.2	8.2	8.3	8.3	8.4	8.4	8.5	8.5	8.6	8.6	8.7	8.7	8.8	8.8	8.9	8.9	9.0	9.0	9.1	9.1	9.2	9.2	9.3	9.3	9.4	9.4	9.5	9.5	9.6	9.6	9.7	9.7	9.8	9.8	9.9	9.9	10.0	10.0
E48	1.0	1.05	1.10	1.15	1.21	1.27	1.33	1.40	1.47	1.54	1.62	1.69	1.78	1.87	1.96	2.05	2.15	2.26	2.37	2.49	2.61	2.74	2.87	3.01	3.16	3.32	3.48	3.65	3.83	4.02	4.22	4.42	4.64	4.87	5.11	5.36	5.62	5.90	6.19	6.49	6.81	7.15	7.50	7.87	8.25	8.66	9.09	9.53																																																																																																																																								
Tolerancias de las series: E6 20% - E12 10% - E24 5% - E48 2%																																																																																																																																																																																								
Valores de las resistencias en Ω , K, M IEC = Comisión eléctrica Internacional																																																																																																																																																																																								

4 Band Resistor Color Code Layout



▼

E-12 E-24 E-48

SERIE E-12

Valor: 789

Buscar Estandar

Inferior: 680 - +

Superior: 820 - +

☰

Lab - Resis

▼

A B C D

A: Gris
B: Negro
C: Rojo
D: Dorado

R= 8 KOhm +- 5%
norm.E24: 8.2 KOhm

Calc colores sabiendo R

☰

▼

Introducir R

Valor de R (Ohm):
8000

OK CANCEL

D: Dorado

R= 0

Calc colores sabiendo R

☰

▼

A B C D

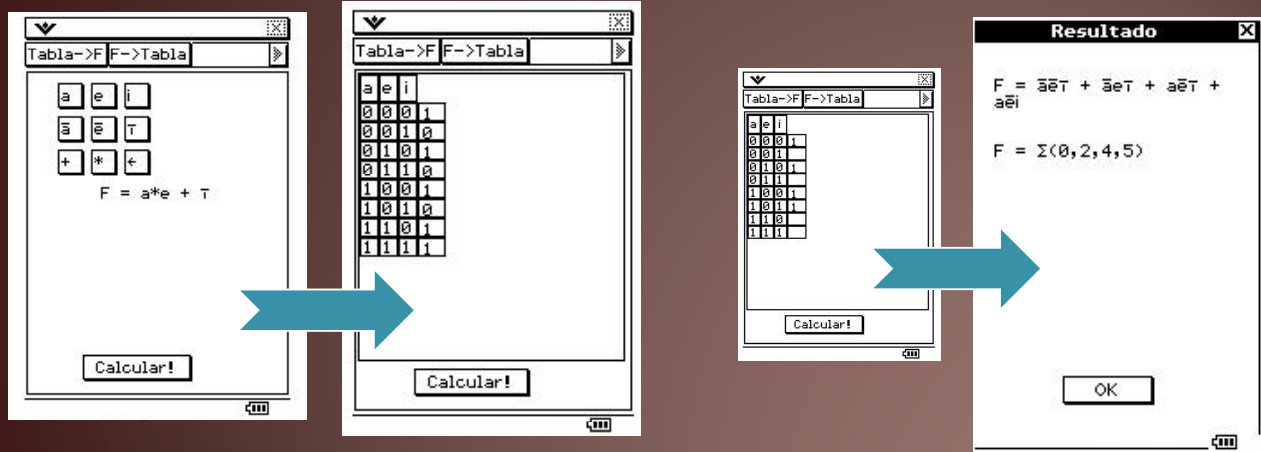
A: Marron
B: Negro
C: Naranja
D: Dorado

R= 10 KOhm +- 5%
norm.E24: 10 KOhm

Calc colores sabiendo R

☰

Lab – Logic Functions



The image shows a sequence of three calculator screens illustrating the calculation of a logic function.

Screen 1: The input expression is $F = a \cdot e + T$. The calculator shows the expression and a "Calcular!" button.

Screen 2: The calculator displays the truth table for the function F :

a	e	T	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Screen 3: The calculator displays the result of the calculation:

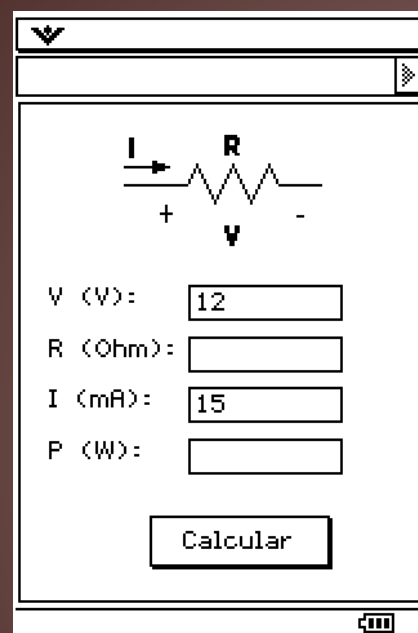
$$F = \overline{a}eT + \overline{a}e\overline{T} + a\overline{e}T + a\overline{e}\overline{T}$$

$$F = \Sigma(0, 2, 4, 5)$$

Check – Ohm's Law

$$V = R \cdot I$$

$$P = V \cdot I$$



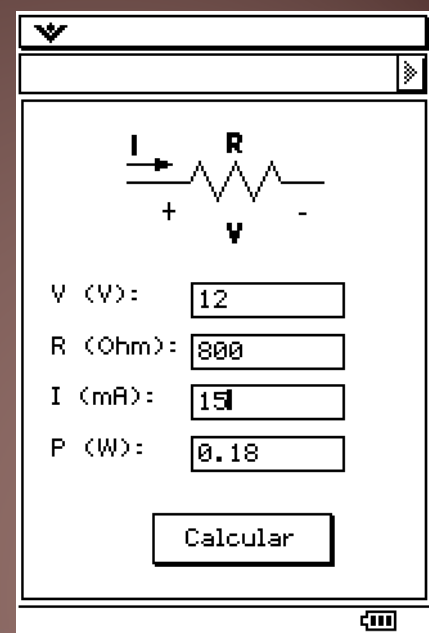
The image shows a calculator screen displaying a circuit diagram and the calculation of power P using Ohm's Law.

Circuit Diagram: A resistor R is shown with current I flowing through it. The voltage V is measured across the resistor.

Calculator Input:

- V (V): 12
- R (Ohm): 800
- I (mA): 15
- P (W): 0.18

The "Calcular" button is visible at the bottom.



The image shows a calculator screen displaying a circuit diagram and the calculation of power P using Ohm's Law.

Circuit Diagram: A resistor R is shown with current I flowing through it. The voltage V is measured across the resistor.

Calculator Input:

- V (V): 12
- R (Ohm): 800
- I (mA): 15
- P (W): 0.18

The "Calcular" button is visible at the bottom.

Check – Tx. Lines

Diagram: A transmission line of length d is connected to a load $R_c + jX_c$. The characteristic impedance is Z_0 and the frequency is f .

Inputs:

- $d(\text{cm})$: 10
- $f(\text{Mhz})$: 3
- $Z_0(\Omega)$: 50
- R_c : 75
- X_c : 25

Calculator!

Results:

- $\lambda = 100 \text{ m}$
- $\beta = 0.0628 \text{ rad/m}$
- $Z_c = 75 + 25j \Omega$
- $Y_c = 0.012 - 4E-3j \text{ S}$

Diagram: A transmission line of length d is connected to a load $R_c + jX_c$. The characteristic impedance is Z_0 and the frequency is f .

Inputs:

- $d(\text{cm})$: 10
- $f(\text{Mhz})$: 3
- $Z_0(\Omega)$: 50
- R_c : 75
- X_c : 25

Calculator!

Results:

- $Y_c = 0.012 - 4E-3j \text{ S}$
- $Z_e = 75.5 + 24.7j \Omega$
- $Y_e = 0.012 - 3.91E-3j \text{ S}$
- $T_c = 0.231 + 0.154j$

Diagram: A transmission line of length d is connected to a load $R_c + jX_c$. The characteristic impedance is Z_0 and the frequency is f .

Inputs:

- $d(\text{cm})$: 10
- $f(\text{Mhz})$: 3
- $Z_0(\Omega)$: 50
- R_c : 75
- X_c : 25

Calculator!

Results:

- $T_c = 0.231 + 0.154j$
- $T_e = 0.233 + 0.151j$
- $\text{SWR}_c = 1.77$
- $\text{SWR}_e = 1.77$

Check - Communications

Diagram: A transmission line with characteristic impedance λ is connected to a load μ .

Inputs:

- λ : 1.5
- μ : 3.5

Calculator

Results:

- $\rho = 0.4285714286$
- $N = 0.75$
- $N_q = 0.3214285714$
- $N_s = 0.4285714286$
- $\text{Var}(N) = 1.3125$
- $T = 0.5$
- $T_w = 0.2142857143$

Diagram: A transmission line with characteristic impedance λ is connected to a load μ .

Inputs:

- $\text{CIR}(\text{dB})$: 10
- $\text{coef}(\alpha)$: 4
- $\text{Sect.}(\Omega)$: 360

Calculator!

Resultado

Results:

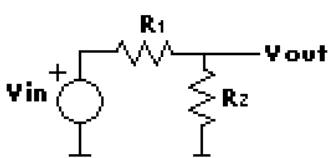
- $K_{\min} = 4.770760687$
- $K = 7$
- $\text{CIR} = 27.45555647$
- $\text{CIR}(\text{dB}) = 14.3863025$

OK

Calculator!

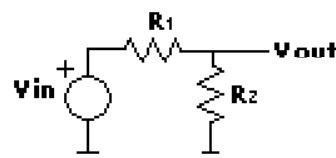
Circ. Analysis - Vdiv

$$V_o = R_2 / (R_1 + R_2) \cdot V_i$$



V_{in} :
 V_{out} :

R_1 : 750 Ω
 R_2 : 150 Ω



V_{in} :
 V_{out} :

R_1 : 1150 Ω
 R_2 : 226 Ω

Circ. Analysis - D-Star

$$R_1 = (R_a \times R_c) / (R_a + R_b + R_c)$$

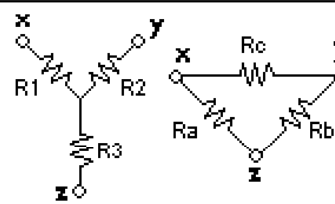
$$R_2 = (R_b \times R_c) / (R_a + R_b + R_c)$$

$$R_3 = (R_a \times R_b) / (R_a + R_b + R_c)$$

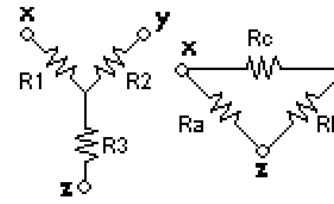
$$R_a = [(R_1 \times R_2) + (R_1 \times R_3) + (R_2 \times R_3)] / R_2$$

$$R_b = [(R_1 \times R_2) + (R_1 \times R_3) + (R_2 \times R_3)] / R_1$$

$$R_c = [(R_1 \times R_2) + (R_1 \times R_3) + (R_2 \times R_3)] / R_3$$



R_1 : R_a :
 R_2 : R_b :
 R_3 : R_c :



R_1 : R_a :
 R_2 : R_b :
 R_3 : R_c :

Contents

1.- Introduction

- C++ vs Basic
- C++ in a calculator

2.- Application structure

- Development environment
- Application model
- Example

3.- Functions

- Embedded functions
- CAS calling

4.- Real application examples

- Laboratory
- Result checking
- Circuit Analysis

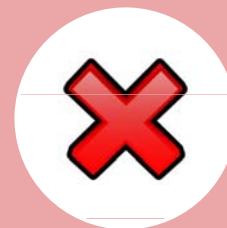
5.- Conclusions

- C++ as future programming language bridging CAS
- Advantages and Drawbacks
- Final statement

4.2- Advantages & Drawbacks



Easy and powerful framework,
versatility and complexity.
Intuitive and useful
applications.



Real and
complex
numbers, new
language (?)



4.3- Final Statement

- C++ is the natural evolution to program calculator add-ins, providing both powerfulness (embedded functions, CAS calling) and simplicity (modular design).
- Web2.0 philosophy: intuitive applications, easy to share and run and capable of being improved by the community.



Suggestions and comments are welcome

Thank you for you attention