CRBasic Programming

Kevin Wang[1], Matthew Conley[2]

[1] Kansas State University
[2] USDA/Arid-Land Agricultural Research Center

Proper Program Structure

```
'Declarations

'Define Constants
Const RevDiff = 1
Const Del = 0 'default
Const Integ = 250
Const Mult = 1
Const Offset = 0

'Define public variables
Public RefTemp
Public TC(6)

'Define Units
Units RefTemp = degC
Units TC = DegC



'Define data tables
DataTable(Temp,1,2000)
  DataInterval(0,10,min,10)
  Average(1,RefTemp,FP2,0)
  Average(6,TC(),FP2,0)
EndTable
```

Declare constants

Declare public variables, dimension array, and declare units.

Define data table

Declarations

THE UNIVERSITY OF ARIZONA

USDA

NSF

KANSAS STATE UNIVERSITY

Proper Program Structure

```
'Begin Program
BeginProg

  'Set scan interval
  Scan(1,Sec,3,0)

    'Measurements
    PanelTemp(RefTemp,250)
    TCDiff(TC()...Offset)

    'Processing (None in this
    'example)

    'Call data table
    CallTable Temp

    'Controls (None in this
    'example)

  'Loop to next scan
  NextScan

'End Program
EndProg
```

Measure

Call data table

Scan loop

THE UNIVERSITY OF ARIZONA

USDA

NSF

KANSAS STATE UNIVERSITY

Basic Elements of CRBasic Program

- **Variables**: named packets of CR1000 memory to store values that normally vary during program execution
  - have alphanumeric names
  - can be dimensioned into arrays of related data
- **Constants**: packets of CR1000 memory to store specific values that do not vary during program executions
  - have alphanumeric names
  - Assigned values at the beginning declarations

Note: Avoid keywords and predefined constants

Basic Elements of CRBasic Program

- **Common instructions**
  - Logic and mathematical operators
  - Program control structures
- **Special instructions**
  - Measurement
  - Control

Declaration - Constants

```
Public PTempC, PTempF
Const CtoF_Mult = 1.8
Const CtoF_Offset = 32

BeginProg
  Scan(1,Sec,0,0)
    PanelTemp(PTempC,250)
    PTempF = PTempC * CtoF_Mult + CtoF_Offset
  NextScan
EndProg
```

Note: 250 ms integration time is used to smoothen the temperature

Declaration - Constants

Table 8. Formats for Entering Numbers in CRBasic		
Format	***Example***	***Base-10 Equivalent Value***
Standard	6.832	6.832
Scientific notation	5.67E-8	$5.67X10^{-8}$
Binary	&B1101	13
Hexadecimal	&HFF	255

Declaration - Variables

```
'Float Variable Examples
Public Z
Public X As Float

'Long Variable Example
Public CR1000Time As Long
Public PosCounter As Long
Public PosNegCounter As Long

Boolean Variable Examples
Public Switches(8) As Boolean
Public FLAGS(16) As Boolean

'String Variable Example
Public FirstName As String * 16 'allows a string up to 16 characters long
```

Note: Default data type: Float

Declaration - Variables

Data Types					
Code	**Data Format**	**Where Used**	**Word Size**	**Range**	**Resolution**
FP2	Campbell Sci. Floating Point	Output Data Supply Only	2 bytes	±7999	13 bits (about 4 digits)
IEEE4 or FLOAT	IEEE 4 Byte Floating Point	Output Data Storage/Variables	4 bytes	±1.4×10E-45 to ±3.4×10E+39	24 bits (about 7 digits)
LONG	4 Byte Signed integer	Output Data Storage/Variables	4 bytes	-2,147,483,648 to +2,147,483,647	1 bit (1)
UINT2	2 Byte Unsigned Integer	Output Data Storage Only	2 bytes	0 to 65535	1 bit (1)
BOOLEAN	4 byte Signed Integer	Output Data Storage/Variables	4 bytes	0, -1	Ture or False (-1 or 0)
BOOL8	1 Byte Boolean	Output Data Storage Only	1 byte	0, -1	Ture or False (-1 or 0)
NSEC	Time Stamp	Output Data Storage Only	8 bytes	seconds since 1990	nanoseconds
STRING	ASCII String	Output Data Storage/Variables	Set by program		

Declaration – Variable Initialization

```
Public aaa As Long = 1
Public bbb(2) As String *20 = {"String_1", "String_2"}
Public ccc As Boolean = True

'Initialize variable ddd elements 1,1 1,2 1,3 & 2,1.
'Elements (2,2) and (2,3) default to zero.
Dim ddd(2,3)= {1.1, 1.2, 1.3, 2.1}

'Initialize variable eee
Dim eee = 1.5
```

Notes:

1. By default (if not assigned values), variables are initially set to zero;
2. **Public** variables can be viewed through the external keyboard/display or software numeric monitors. **Dim** variables cannot.

Declaration – Alias and Unit

- A variable can be assigned a second name, or alias, by which it can be called throughout the program.
- Units are not used elsewhere in programming, but add meanings to resultant data table headers.

```
Alias var_array(1) = solar_radiation
Alias var_array(2) = quanta

Units solar_radiation = Wm-2
Units variable2 = moles_m-2_s-1
```

Declaration – Data Table

```
DataTable(TableName,True,-1)
  'FP2 Data Storage Example
  Sample(1,Z,FP2)

  'IEEE4 / Float Data Storage Example
  Sample(1,X,IEEE4)

  'UINT2 Data Storage Example
  Sample(1,PosCounter,UINT2)

  'LONG Data Storage Example
  Sample(1,PosNegCounter,Long)

  'STRING Data Storage Example
  Sample(1,FirstName,String)

  'BOOLEAN Data Storage Example
  Sample(8,Switches(),Boolean)

  'BOOL8 Data Storage Example
  Sample(2,FLAGS(),Bool8)

  'NSEC Data Storage Example
  Sample(1,CR1000Time,Nsec)
EndTable
```

Notes for "DateTable":
"TableName" – data table name
"Ture" – always run the table
"-1" – divide the available memory to all tables

Notes for "Sample":
"1" – 1 sample to store
"Z" – sample saved in variable Z
"FP2" – Data type of Z is FP2

Declaration – Data Table

```
'Define Data Tables
DataTable(OneMin,True,-1)
  DataInterval(0,1,Min,10)
  Average(1,Batt_Volt,FP2,False)
  Average(1,PTemp_C,FP2,False)
  Average(2,Temp_C(1),FP2,False)
EndTable

DataTable(Table1,True,-1)
  DataInterval(0,1440,Min,0)
  Minimum(1,Batt_Volt,FP2,False,False)
EndTable
```

Notes for "DateTable":
"OneMin" – data table name
"Ture" – always run the table
"-1" – divide the available memory to all tables

Notes for "DataInterval":
"5" – data storage occurs at 5 minutes into every hour,
"60" – "Data storage occurs every 60 (min)
"Min" – Data table occurs every (60) minute
"10" – up to 10 lapse time stamps will be recorded

Notes for "Average":
"2" – 2 averages are calculated,
"Temp_C(1)" – Calculation starts from Temp_C(1),
"FP2" – Save the averages in FP2 format
"False" – Do not disable (do the averages)

Program Control Structures

- Scan … Next Scan
  - Establishes the program scan rate. **ExitScan** and **ContinueScan** are optional.

```
Syntax
Scan(Interval, Units, Option, Count)
  [statement block]
ExitScan
  [statement block]
ContinueScan
  [statement block]
NextScan
```

Program Control Structures

- If … Then … ElseIf … Then … Else … EndIf
  - Allows conditional execution, based on the evaluation of an expression. **Else** is optional. **ElseIf** is optional.

Syntax

```
If [condition] Then [thenstatements] Else [elsestatements]
```

-or-

```
If [condition 1] Then
  [then statements]
ElseIf [condition 2] Then
  [elseif then statements]
Else
  [else statements]
EndIf
```

Program Control Structures

- Do … Loop
  - Repeats a block of statements while a condition is true or until a condition becomes true.

Syntax

```
Do [{While | Until} condition]
  [statementblock]
[ExitDo]
  [statementblock]
Loop

-or-

Do
  [statementblock]
[ExitDo]
  [statementblock]
Loop [{While | Until} condition]
```

Note: The condition is checked at the beginning of the loop – the loop may not be executed if the condition is not met.

Note: The condition is checked at the end of the loop – the loop will be executed at least once.

Program Control Structures

- Subroutines
  - Declares the name, variables, and code that form a Subroutine. **Argument list** is optional. **ExitSub** is optional.

Syntax

```
Sub subname (argument list)
  [statement block]
Exit Sub
  [statement block]
End Sub
```

Measurement instructions

- VoltDiff
  - Measures the voltage difference between **H** and **L** inputs of a differential channel.

```
Syntax
    VoltDiff(Dest, Reps, Range, DiffChan, RevDiff, SettlingTime,
        Integ, Mult, Offset)
```

- VoltSe
  - Measures the voltage at a single-ended input with respect to ground.

```
Syntax
    VoltSe(Dest, Reps, Range, SEChan, MeasOfs, SettlingTime,
        Integ, Mult, Offset)
```

Input instructions

- SerialOpen
  - Sets up a datalogger port for communication with a serial port.

```
Syntax
    SerialOpen(ComPort, BaudRate, Format, TXDelay, BufferSize)
```

- SerialInBlock
  - Stores incoming serial data. This function returns the number of bytes received.

```
Syntax
    SerialInBlock(ComPort, Dest, MaxNumberBytes)
```

Reference

- Operator's Manual, CR3000 Measurement and Control System, Revision: 5/13. 2013. Campbell Scientific, Inc.
- http://s.campbellsci.com/documents/us/manuals/cr3000.pdf

Lecture presented at:
Field Phenomics Workshop
Maricopa Agricultural Center
Maricopa, AZ
March 16-19, 2015

For further material relating to field phenomics and information on future workshops, visit **fieldphenomics.org**

THE UNIVERSITY OF ARIZONA

USDA

NSF

KANSAS STATE UNIVERSITY