

Computer-Aided Analysis of Mechanisms Using ChExcel

**Kabileshkumar G Cheetancheri
Harry H. Cheng**

**Integration Engineering Laboratory
Department of Mechanical and Aeronautical Engineering
University of California
Davis, California, 95616
Email: hhcheng@ucdavis.edu**

Copyright © September, 2005, All rights reserved

Abstract

As computer CPU gets fast, script computing is increasingly popular. Script computing allows a computer program to be executed interpretively without tedious compilation and linking. It can be used for rapid prototyping, embedded scripting, mobile computing, teaching, and learning.

Ch is an embeddable C/C++ interpreter. It execute C/C++ programs as scripts interpretively without compilation. In this project, Ch and Ch Excel are used for mechanism design and analysis. Microsoft Excel is a user friendly software commonly used by engineers to solve many computational problems. Using Excel as a GUI front-end, many Excel features are readily available for applications and use Ch as backend to handle computations. A familiar structure programming paradigm in C can be used for scripts with capabilities of other software modules in Ch. Mechanism design and analysis modules developed in Ch, Ch Excel and Ch Mechanism Toolkit are ideal solutions with user friendly Excel interface for solving mechanism design and analysis problems.

Contents

1	Introduction	3
1.1	Compiler Vs Interpreter	3
1.2	Software and Mechanisms	4
1.3	Ch and Mechanism Toolkit	4
1.4	Spreadsheet	5
1.5	Scripting	5
2	Design and Analysis of Fourbar Linkage Using Ch and ChExcel	7
2.1	Ch Excel	7
2.1.1	Plotting Variety	7
2.2	Ch Mechanism ToolKit	8
2.2.1	Kinematic Analysis	8
2.2.1.1	Angular Position	9
2.2.1.2	Angular Velocity	9
2.2.1.3	Angular Acceleration	9
2.2.1.4	Dynamic Force Analysis	9
2.2.1.5	Animation	10
2.3	Excel Worksheet	10
2.3.1	Development of Excel sheet	10
2.3.2	Description of Excel Work Sheet	11
2.3.3	Processing of Excel Work Sheet	11
2.3.4	Execution of fourbar_position_analysis.xls	13
2.4	Position Analysis	14
2.4.1	Initialization	14
2.4.2	Feasibility of Fourbar Linkage Formation	15
2.4.3	Angular Position	15
2.4.3.1	Range of Motion	16
2.5	Animation	16
2.6	Plotting	18
2.7	Angular Velocity Analysis	20
2.8	Angular Acceleration Analysis	23
2.9	Dynamic Analysis	24

3	Analysis of Other Mechanism Using Ch and ChExcel	29
3.1	Position Analysis of Geared Fivebar Mechanism	29
3.2	Slider Position Analysis of Crank Slider Mechanism	31
3.3	Position Analysis of Multi-Loop Six Bar Linkages	33
3.3.1	Position Analysis of Fourbar Slider Mechanism	33
3.3.2	Position Analysis of Watt Six-Bar(I) linkage	36
3.3.3	Position Analysis of Watt Six-bar (II) Linkage	39
3.3.4	Stephenson Six-bar (I) Linkage	42
3.3.5	Stephenson Six-bar (III) Linkage	45
4	Conclusions	49

Chapter 1

Introduction

Today's global economy has raised the level of competition throughout the world. Globalisation removed physical boundaries between countries leading to companies venturing all over the world. This increased level of competition has forced companies to be lean, adaptable and responsive while maintaining the quality in order to stay in the market. The penetration of information technology into scientific field has led to development of various application specific software. This has led companies to look for more diversified people which made instructors to teach students with a variety of software and their application, but one of the major issue is choosing the environment in which students can get quick start in programming.

Ch [1][2] was designed and implemented for novice and inexperienced users to get start with programming. Ch is a high-level language environment, it allows students to focus on program structure and algorithm instead of tedious compile link distraction. Due to the availability of various softwares specialising in particular field, students and researchers may find it difficult to procure different software for each application. This could prevent them in using the latest technology which can improve their efficiency. Ch environment could become a useful tool in this scenario, where the software modules that has been integrated with this environment would prove helpful for engineers and scientist for learning and research purpose.

1.1 Compiler Vs Interpreter

The process of execution of a program marks the difference between a compiler and interpreter. Compiler translates a source program to a binary module representing the program in machine readable format. The compiled program is then linked to obtain the executable format. Where as an interpreter interprets the source program line by line, it takes up the input data and the source program at the same time and do not produce any executable files. In case of interpreter, the source code is translated only if it is executed and the error messages are tied to the source which make debugging easier and also working environment more supportive. An interpreter is used when filters are implemented in the operating system kernel where compilation may not be feasible due to protection and robustness concerns even though compilation increases runtime efficiency. Interpreters increase the execution over head since the program is continuously re-examined, and also increase space overhead compared to compiler. But considering the fact that the speed of the processor and the storage capacity has been growing at such a pace that the execution overhead

and space overhead can be neglected to some extent. The usage of interpreter by novice software developers who may have to frequently modify the code is a good choice as they don't have to compile every time they debug. Techniques like self-modifying code is easy to implement using interpreter and is also conceptually simpler and could be of great help to beginners.

1.2 Software and Mechanisms

The necessity of computer technology in every field ranging from literature to space technology has been proved beyond doubt. It is so advanced and diversified that every field needs a specific software to perform the tasks efficiently. Computational methods involved in mechanism design and analysis have led to development of various application software that help engineers to design complicated mechanisms. Packages like Automated Dynamic Analysis of Mechanical Systems (ADAMS) were developed to solve complicated engineering design and analysis problems. There are various special purpose software packages such as WATT by Heron[3], Pro/Engineer simulation[4], Simulation and Analysis of Mechanisms [5] which provides powerful and easy tools to draw, design, analyze motion and force of arbitrary 2D-mechanisms. Compared to various general purpose software available, these specialized packages are easier to use to obtain the desired result. But from a learner's point of view, learning how computations are performed from a menu driven software is always going to be challenging. It is necessary that the students learn the background of this software development and to understand how the computational aspects are applied in software to obtain the solutions. There is also a possibility that these commercially available package and software does not allow users to incorporate their special needs. In that case, the user may have no option to write their own software module to solve their unique problem. The mechanism toolkit[6] fills up this gap.

1.3 Ch and Mechanism Toolkit

Ch [1] was originally developed by Cheng. Ch was designed as a superset of C programming language extended for numerical computing, network computing, distance learning and interpretive programming. Ch language environment has other software modules integrated to its environment using tools like Embedded Ch [7] and Ch SDK [8]. By using these tools various software [9] has been integrated to this environment. For example, OpenGL [10], OpenAL [11], XML [12] were integrated using ChSDK. The goal is to make Ch Language environment as a single destination for heavily used softwares.

The Ch Mechanism Toolkit[6] is significantly different from other packages in terms of providing the source code to the user. The users now have the freedom to modify the available code, according to their requirements, which is provided with Ch Mechanism Toolkit. By analyzing the source code, the user can develop their own module to solve complicated engineering design and analysis problems which may be unique to their field or experiment.

Unlike other mathematical software packages, Ch conforms to the open C/C++ standard. The various high-level mathematical features of Ch such as complex numbers, matrix operations are very useful in engineering applications. The availability of various other softwares that can be executed in Ch environment makes it more comfortable for the user to use the best software for a

given application. For example, Ch gives easy solution for mechanism animation, it could be the ideal environment to develop mechanism related animation. The user specific development in Ch can be done by adding small software module by the user, which serves their personal requirement. Once it has been inducted it remains in the environment for ever and can be used repetitively. This makes the task of developing and maintaining the software as simple as possible. Even though this mechanism toolkit has been developed for simple mechanisms, this can be expanded to other complicated mechanisms.

Ch also has a well built toolkit to deal with control system design and analysis [13][14]. The development of modules of classes in this toolkit [14] makes it easier to develop a new control system. Ch can be used for CGI[15] which allows the user to access control [14] and mechanism [6] toolkit through the web. This mechanism toolkit is implemented as a module which can be utilized from various other softwares that has been embedded with Ch. ChExcel [16] is one among the newly developed application by which Ch can be accessed from Excel. In this work the utilization of this mechanism toolkit with Ch Excel has been demonstrated. Availability of demo programs along with this package helps novice designer to understand the process of using software in solving mechanical problems. They can also modify these programs to solve their own mechanism design and analysis problems. Chapters 2 and 3 explain in detail on how ChExcel has been used in analyzing various mechanisms.

1.4 Spreadsheet

Spreadsheet was originally created to help the accountants in their business applications. It was later used in educational purpose and is now being widely used in almost every field. More recently, spreadsheet was used to simulate engineering system, such as logic networks, control systems and antenna array design [17]. The spreadsheet modules has been used in the field of machine design[18]. Spreadsheet is also used to aid the designer in the approximation process [19]. There are various application of spreadsheet modules like design of worm gear geometry [20], determination of spur gear form factors [21] and various other applications. These research projects highlights the usage of Excel in analyzing various mechanisms.

1.5 Scripting

Most software systems are seldom developed starting from scratch, but rather built using the existing libraries [22] which led to recent sprout in development of scripting languages [23]. To have various advantages over a compiled language a scripting language must provide two capabilities. First, developers must have a large number of software components readily available to be used in scripts. These components must support many application domains such as graphical user interfaces, internet communication or text processing [23]. Secondly, the scripting language environment must integrate seamlessly with software components and applications written in system programming languages. These advantages can be seen in Ch script, where a Ch script can access all the software that has been integrated to the Ch language environment and also integrate seamlessly with other software component.

This research project presents the usage of ChExcel for analyzing various mechanisms. Excel

spreadsheets were developed for specific purpose which utilize Ch Mechanism Toolkit in back end to analyze the mechanisms. It also exposes various other software integrated to Ch to be accessed from Excel. The second part of this research project is focused on integration of Java with Ch for Windows version and Kaffe for Linux. The final chapter illustrates application of ChJava in manipulating a mechatronic system in stand alone and in web-based application(client side and remote).

Chapter 2

Design and Analysis of Fourbar Linkage Using Ch and ChExcel

Computers have changed the way design and research are being performed. They are so sophisticated that decision regarding development of new prototypes are taken only after getting satisfying results on running simulations. The use of software for applications makes the process of research and development easier and more efficient. This chapter focuses on utilizing a spreadsheet in analyzing various mechanisms using ChExcel.

2.1 Ch Excel

Ch Excel was designed and developed to use the computational power of Ch from Excel worksheets [16]. The availability of Add-In technique in Excel gives a seamless extension from Excel to Ch. Students and researcher's who have been using Microsoft Excel to great extent would find a programming language like Ch integrated with Excel can increase its usability many folds. For example, various user defined software modules that have been developed in Ch can be accessed from the worksheets. Excel being one of the widely used spreadsheet by engineers, adapting to new addition with Excel shall not be a difficult task for the user when Excel is used as front end while Ch does the computation at the back end [24].

The simple GUI front end in assembling the input and output cells makes interaction with the user easier. Excel's ability to check the values entered in a cell is utilized in this work for preliminary decision making and the complicated issues are handled by Ch running in the backend. As a general purpose programming language, the amount and the variety of work that Ch Excel could perform are almost unlimited.

2.1.1 Plotting Variety

Excel is mainly used in tabulating data which lead to plotting of graphs. Adding to the variety of graphical options already available in Excel, Ch provides a wider variety of 2D/3D graphical plotting [2], which can be saved in variety of formats. Availability of different format of graphs prevents the user to change other features to fit the given graph format. For example, to publish a graph using Latex, the user may need a graph in postscript format which can be generated from

Ch. This capability would play an important role when several people exchange information for their research.

2.2 Ch Mechanism Toolkit

The Ch mechanism toolkit [25][6] consists of many C++ style classes, namely `CFourbar`, `CCrankslider` and `CGearedFivebar`, which can be used to handle the analysis of various planar mechanisms like fourbar, crank-slider and geared-fivebar linkages respectively. The input is obtained from the Excel spreadsheet and using Ch Mechanism Toolkit in back end, various positions, velocity, acceleration of different links are determined. The output is then displayed according to the user specification on the same spreadsheet through which input is given. Classes in Ch Mechanism Toolkit have several member functions for calculating the angular positions, velocities and accelerations of individual link members of the specified mechanism configuration.

2.2.1 Kinematic Analysis

The simplest closed-loop planar mechanism is the fourbar linkage shown in Figure 2.1. The kinematic analysis of the linkage mechanism is performed by functions defined in the `CFourbar` class. Some of the functions are described in the following sections.

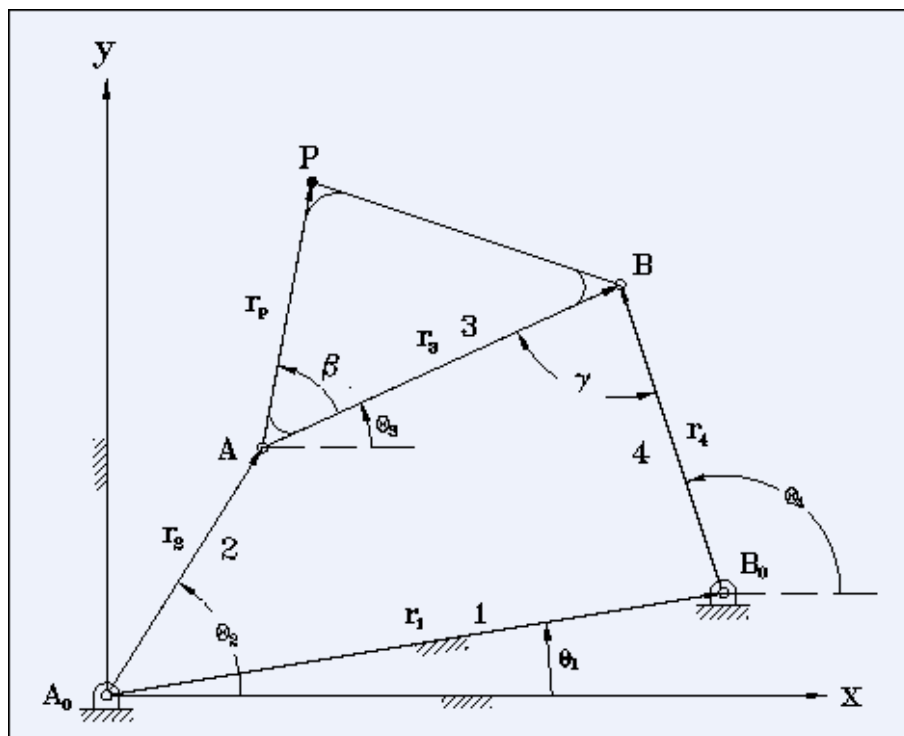


Figure 2.1: Fourbar Mechanism.

2.2.1.1 Angular Position

Given the link lengths of a four-bar linkages along with θ_1 and θ_2 , the angular positions θ_3 and θ_4 can be solved using the function `complexsolve()`. In general, there are two sets of solutions of θ_3 and θ_4 for a given θ_2 , which corresponds to the two different circuit or two geometric inversions. The function `angularPos` in the class `CFourbar` is used in determining the two set of solutions.

```
CFourbar::angularPos(theta_1, theta_2, theta_id)
```

Where `theta_1` and `theta_2` are arrays which gets the two sets of solution returned from the function.

2.2.1.2 Angular Velocity

The velocity analysis for a closed loop linkage can be carried out from its loop-closure equation. Given the lengths of all links and angles θ_1 and θ_2 the angular velocities of link 3 and link 4 can be found out using the function `angularVel`

```
CFourbar::angularVel(theta, omega, omega_id)
```

The other required angles can be found using the `angularPos` function described earlier. `theta` and `omega` are arrays which contains the angle that every link makes with ground and the known angular velocity of the link. `omega_id` represents the link number whose angular velocity is known.

2.2.1.3 Angular Acceleration

The acceleration is the second derivative of the loop closure equation. A fourbar class member function `angularAccel()` calculates α_3 and α_4 when all the angle that the links make with ground and their angular velocity along with angular acceleration of link 2 is given.

```
CFourbar:: angularAccel(theta, omega, alpha, alpha_id)
```

The variables `theta` and `omega` have the angle and the angular velocity of the links in the mechanism. The variable `alpha` has the angular acceleration of the known link and its corresponding link number is sent to the function through the variable `alpha_id`.

2.2.1.4 Dynamic Force Analysis

Dynamic force analysis is performed based on D'Alembert's principle. In this analysis we assume that the position, velocity and acceleration of each moving body in the system is already known. The user can also use the above explained methods to find the required velocity and acceleration of each link. The method `forceTorque` defined in the `CFourbar` class, using which the forces acting on those links can be identified.

```
CFourbar::forceTorque(theta, omega, alpha, t1, X)
```

where `t1` is the input torque and `X` is the output array of all reaction forces that has been calculated in this function.

2.2.1.5 Animation

One of the best advantages of using ChExcel is that the developer can use all the software modules that have been integrated to Ch. For example, `qanimate` which can be accessed from Ch can be used for developing a mechanism animation. This may not be possible if some one uses only Excel to analyze the mechanism. This animation is made possible by a method called `animation()` which is defined in the class `CFourbar` as `CFourbar::animation(circuit_no)`.

2.3 Excel Worksheet

2.3.1 Development of Excel sheet

The input requirements for a particular problem are first defined to develop the excel sheet. It should also be noted that each cell is unique and should not be used for multiple usage. The cells that takes up the input values are arranged in the same order as the values are to be entered. Corresponding comments that appear in the first column refers to the values which are either input or output for the given problem. The user can go through all the cells that requires input data and enter the data. The automatic calculation is over-ridden by ChExcel, so the user may not hesitate to go through the cells, only after pressing F2 any function indicated in the cell would execute, whereas by default in Excel pressing `enter` key would invoke the process. A variable used in computation has to be declared first. This is done by using function **ChRunScript()**. This performs the role of a destructor and constructor. It removes all previously declared variables first and initializes the variables that would be used in the current context in Ch program. The function **ChPutMatrix()** takes the input value from Excel.

The same variables that handle the values obtained from Excel are used in computing the result in Ch. These computation occurs in the background using the Ch computation power. As Ch has been integrated with various other software modules, the developer can use the best available software to perform the required task. As Ch has all capabilities of Matlab, input values can be treated as a computational array. This quality of Ch is quite handy while using Excel, as every input is in terms of columns and rows which can be considered as matrix. Availability of pre-written functions in Ch to deal with matrix makes coding easier for handling matrix operation. For example, calculating the inverse of a matrix is done just by calling `inverse(A)` function which takes an argument of a two dimensional array and returns its inverse considering the two dimensional array as a matrix with rows and columns.

All the values that is generated from the script file is stored in a Ch variable. From Excel, these variable are called by using function **ChGetMatrix()** [16]. This function gets the values from the Ch script files and are displayed in the designated cells. As the input and output are in terms of matrices, the results are easily distributed among the rows and columns which give clear view of the results obtained. This also reduces the amount of work involved in writing code to print the values in a tabular format. As Ch has variety of software integrated in it, the kind of output that is required by the user can be obtained by using those software modules.

2.3.2 Description of Excel Work Sheet

The variables that are displayed in column A are the input and output of this problem. The variables R1, R2, R3, R4 represent the lengths of the links of a fourbar linkage as shown in Figure 2.1. The angle theta1 is the angle made by the link 1 with the ground and angle beta is the angle that the coupler link makes with the link 3. The known angle cell contains the angular position of the stationary link link1. The other angle which is required to be known could be of the angle that link 2 or link3 or link4 makes with the ground. The cell A15 contains a drop down list of various angle number and the cell A16 is provided with the corresponding angle for calculating the angular position of other links.

The fourbar link has two circuit for a given set of inputs. Therefore, there are two sets of result for the problem as shown in Figure 2.3. The two sections in the spreadsheet namely the `first solution` and the `second solution` represent the two results. The next section in the Excel sheet deals with the limits of the input range and its corresponding output range where the angle covered by link2 represents the input range and the output range represents the minimum and maximum angle that link4 can make with the ground.

2.3.3 Processing of Excel Work Sheet

The communication to Ch has to be restarted using `RestartCh` button for a new process[16]. The user can be given the choice of selecting their required units for the input values. By default, unit for angle is `radians` and for length is `meters`. The Excel sheet is designed to incorporate all input values in the order the user goes through for inputting the values. Preliminary validations are performed using Excel's built in functions. This makes writing a program in the backend a lot easier, by eliminating the conditional statements that may require to validate the input values and also reduce the time taken.

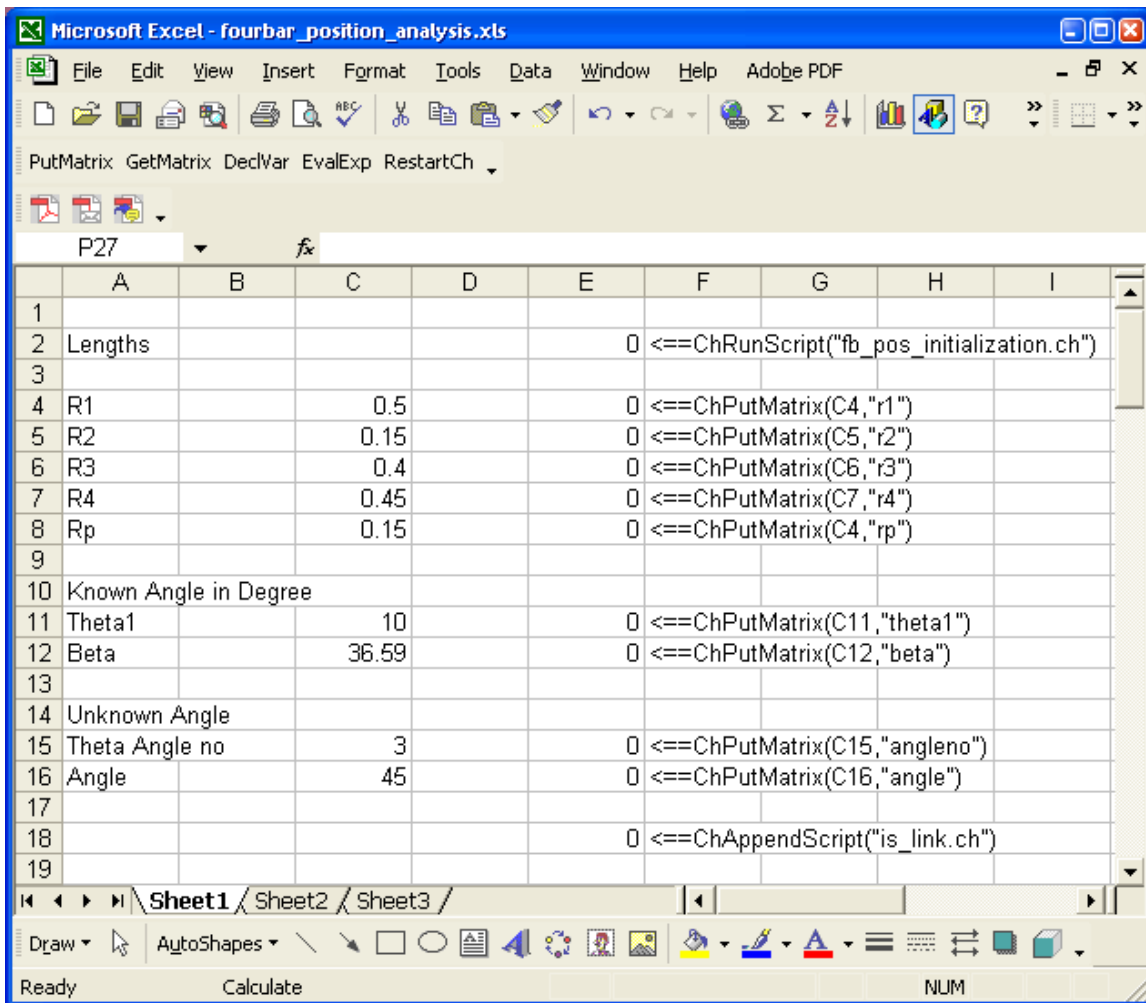


Figure 2.2: ChExcel spreadsheet of fourbar linkage position analysis (fourbar_position_analysis.xls).

To call the Ch scripts from Excel, certain cells have been inducted with Ch API's. The script file has to be executed in a predetermined sequence as script files used depend on the values determined by the previous script files. This being the case, the script files are arranged in the required order it has to be called. The user has to go through the cells one by one in column E in this example. To initiate the call the user has to press F2 which activates the command that executes the script file inducted in column E [16]. On execution of these function, depending up on its success or failure, the corresponding cells in coloumn E are returned with values either **0** upon success or **#ERROR#** upon failure.

All the values that have been calculated in Ch are stored in a variable which are then called from Excel when it is needed. As all the variables are global, the values can be used by different script files. As the values are handled in terms of matrix, the results are easily distributed among the rows and columns as shown in Figure 2.3.

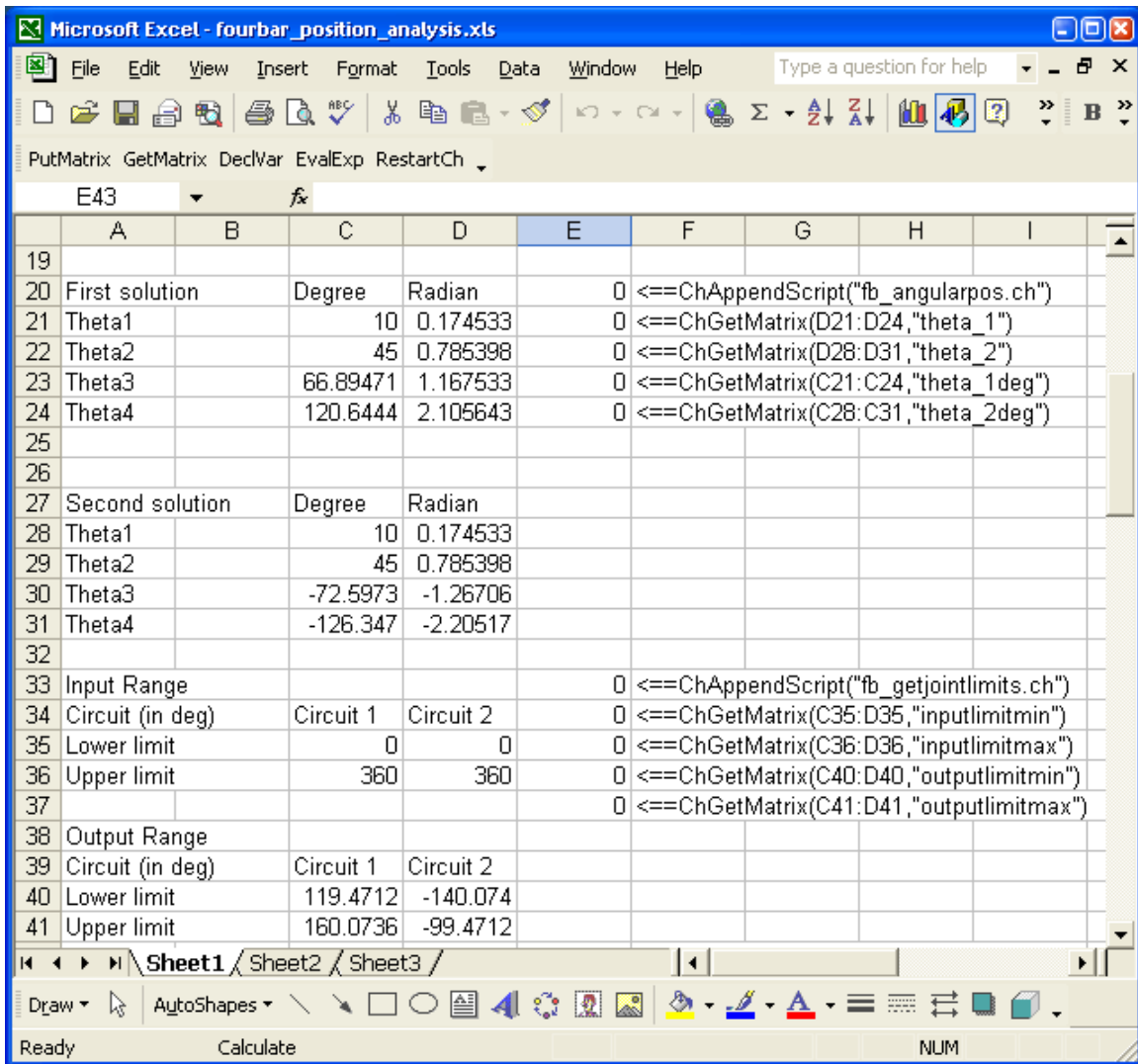


Figure 2.3: ChExcel spreadsheet of fourbar linkage position analysis (fourbar_position_analysis.xls) (contd.).

2.3.4 Execution of fourbar_position_analysis.xls

After refreshing the link between Ch and Excel, the script file fb_pos_initialization.ch is executed using the function **ChRunScript()** which is inducted in the cell E2.

```
ChRunScript("fb_pos_initialization.ch")
```

On executing this function, Ch removes all the variables that have been declared before and executes the statement in this file which may declare new variables according to the requirements. Once this has been performed, the input values are to be entered in the respective cells. These values are read from cells by the ChExcel function **ChPutMatrix()**. For example, to read the input value of R1 that has been entered in cell C4, the function **ChPutMatrix(C4, "r1")** is used in declaring and initializing the variable r1 with the value that has been entered in cell C4. This variable r1 can be also used as such in Ch for computing purpose. The length of links are given

in respected cells and is read and stored in the corresponding variable using **ChPutMatrix()** function. On completion of storing all the variables with the required input, script file `is_link()` is executed. This script file is appended to the previous script by **ChAppendScript()**.

```
ChAppendScript("is_link.ch")
```

This file can be called every time the user changes the input contents. Similarly the program `fb_angularpos.ch` is also appended to determine the angular position of the link. The initialization should not be performed in files that are appended, as it does not destroy the previously declared variables. So, there is a possibility of redeclaring a variable which may lead to syntax errors. Once the required calculations are performed, the output is displayed using function **ChGetMatrix()**. For example, to display the output in radians for first solution,

```
ChGetMatrix(D21:D24,"theta_1")
```

which displays `theta_1` in cells D21 through D24. The cells in column E has a value **0** throughout, which states that all the functions were successfully called.

2.4 Position Analysis

This section would explain the various script files used in position analysis of a Fourbar linkage mechanism.

2.4.1 Initialization

The script `fb_pos_initialization.ch` is used in initialization of all the variables. The script includes the header files, declares and initializes variables that will be used. This file does not either declares or initializes the variables that has been declared in Excel because the same variable can be used in both Ch and Excel. For example the variable `theta_1` is declared in Ch but is used in Excel and the variable `r1` is declared and initialized in Excel but can be used as any other variable in Ch. The command **ChPutMatrix(C4,"r1")** when executed initialize the variable `r1` and store the value entered in the cell C4.

```

#include <math.h>
#include <fourbar.h>
#include <chplot.h>
#include <stdio.h>
#include <windows.h>
#define APP_NAME "Link Status"
  PCHAR string="This Link is not Possible\n";
  PCHAR string1="This Link is Possible\n";

double theta_1deg[1:4], theta_2deg[1:4];
double theta_1[1:4], theta_2[1:4];
int fourbartype;
double inputlimitmin[2], inputlimitmax[2],
  outputlimitmin[2], outputlimitmax[2];
int i;
double retcurvex[51], retcurvey[51];
string_t islink;
CFourbar fourbar;
class CPlot plot;

```

Program 2.1: Initialization of variables used in position analysis of fourbar linkage (fb_pos_initialization.ch).

The header file windows.h is included to utilize the message box. This file is called only once before refreshing the link between Ch and Excel. The initialization file is called using the Ch API ChRunScript which acts as both destructor and constructor.

2.4.2 Feasibility of Fourbar Linkage Formation

The script file is_link.ch is used in determining the feasibility of a fourbar linkage formation with the given set of link lengths and angles. This script file will display either This Link is not possible or This Link is possible in a message box which clearly states the status of the problem and user can make the required changes without disconnecting the communication between Ch and Excel. This script uses the function CFourbar::grashof() to determine the status of the proposed link.

```

fourbar.setLinks(r1,r2,r3,r4,theta1);
fourbartype=fourbar.grashof(islink);

if(fourbartype == FOURBAR_INVALID)
{
  MessageBox( NULL, string, APP_NAME, MB_OK | MB_SYSTEMMODAL | MB_NOFOCUS);
}
else
{
  MessageBox( NULL, string1, APP_NAME, MB_OK | MB_SYSTEMMODAL | MB_NOFOCUS);
}

```

Program 2.2: Validation of a fourbar linkage formation(is_link.ch).

2.4.3 Angular Position

Angular position of the links are determined in the script file fb_angularPos.ch as shown in Program 2.3. The input values theta1 and the length of links are obtained from Excel. The function CFourbar::angularPos() determines the angular position of various links. The possible formation of two circuits is also considered and two sets of results are determined. These

sets of two values are stored in two separate variables which are then displayed in Excel. The output is printed in Excel by the function **ChGetMatrix()**, using the same variable that was used in Ch.

```

theta1 = theta1*M_PI/180;
theta_1[1] = theta1;
theta_1[angleno] = angle * M_PI/180;
theta_2[1] = theta1;
theta_2[angleno] = angle * M_PI/180;
beta = beta * M_PI/180;
fourbar.setLinks(r1, r2, r3, r4, theta1);
fourbar.setCouplerPoint(rp, beta);
fourbar.angularPos(theta_1, theta_2, angleno);

for(i=1;i<=4;i++)
{
    theta_1deg[i] = theta_1[i]*180/M_PI;
    theta_2deg[i] = theta_2[i]*180/M_PI;
}

```

Program 2.3: Angular position analysis of fourbar linkage (fb_angularPos.ch).

2.4.3.1 Range of Motion

The angular ranges of the input and output links are determined in this script file. This is done by using the function `CFourbar::getJointLimits()`. This gives the range of the input link and the output link in unit of radian which are later converted to degree explicitly. On analysing the given input details it was found to form two circuits. The ranges for two circuits are calculated using the script file `fb_getJoinLimits.ch` and output of this file is displayed in the corresponding cells on the Excel sheet as shown in Figure 2.3.

```

fourbar.setLinks(r1, r2, r3, r4, theta1);
fourbartype = fourbar.getJointLimits(inputlimitmin, inputlimitmax,
                                     outputlimitmin, outputlimitmax);

for(i = 0; i<=1; i++)
{
    inputlimitmin[i]*= 180./M_PI;
    outputlimitmin[i]*= 180./M_PI;
    inputlimitmax[i]*= 180./M_PI;
    outputlimitmax[i]*= 180./M_PI;
}

```

Program 2.4: The range for motion of fourbar linkage (fb_getJointLimits.ch).

2.5 Animation

Animation feature of Ch was utilized in creating an animation sequence of the designed fourbar linkage. A separate Excel spreadsheet was designed to demonstrate animation for the given set of links as shown in Figure 2.4. The various initialization is performed in the script file `fb_ani_initialization.ch`. This script is invoked by function **ChRunScript(fb_ani_initialization.ch)**, which removes all previously declared variables and creates an array `theta_1`. All the other input values like the length of links, angles are obtained by function **ChPutMatrix()** from the Excel sheet. There are multiple branches for a fourbar linkage. The branch of a fourbar linkage animation is determined by the user using a drop down list in cell C15. The corresponding

branch number is stored in variable `branchno`. Script file `fb_animation.ch` that has been inducted in cell E16 is then executed. The single line in the script that performs the animation is `fourbar.animation(branchno)`, is one of the reason why Ch can be considered as an easy tool for learning purpose. The corresponding animation is displayed in another window as shown in Figure 2.5)

```
#include <math.h>
#include <fourbar.h>

double theta_1[1:4];
CFourbar fourbar;
```

Program 2.5: Initialization for animation of a fourbar linkage (`fb_ani_initialization.ch`).

```
fourbar.setLinks(r1, r2, r3, r4, theta1);
fourbar.setCouplerPoint(rp, beta, TRACE_ON);
fourbar.setNumPoints(50);
fourbar.animation(branchno);
```

Program 2.6: Animation of a fourbar linkage (`fb_animation.ch`).

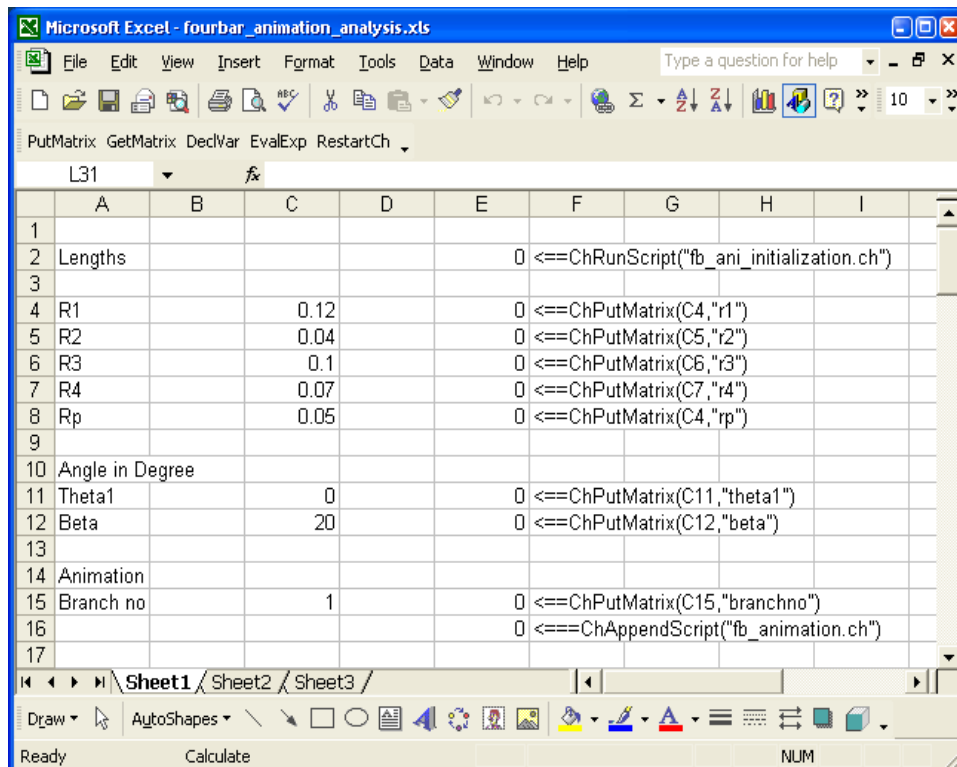


Figure 2.4: ChExcel spreadsheet for fourbar linkage animation. (`fourbar_animation_analysis.xls`).

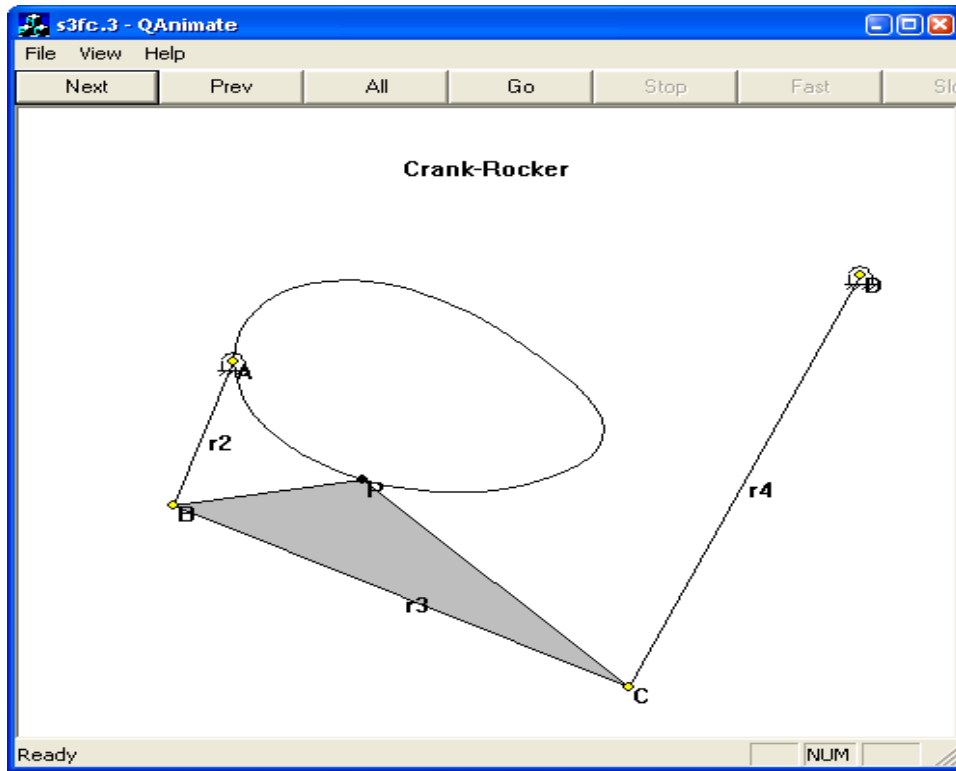


Figure 2.5: Snap shot for a fourbar linkage animation.

2.6 Plotting

Ch has the option of 2D/3D plotting which can also be saved in different format like gif, postscript, png, etc. This feature is utilized in getting the plotting of coupler curve of a fourbar linkage. An Excel spreadsheet was designed and developed to plot the coupler curve as shown in Figure 2.6 which has the comments on column A. The script file `fb_plot_initialization.ch` is invoked by the `ChRunScript()` which does the required initialization. The input values are entered in column C corresponding to the comment in column A, which are stored in variables using ChExcel function `ChPutMatrix()`. As this mechanism has two different branches for given set of input data, the user has been given an option of selecting the branch number from the drop down list from cell C15. On execution of script file `fb_curve.ch`, it displays the selected branch's coupler curve as shown in Figure 2.7. The function that creates plotting is `fourbar.plotCouplerCurve()`, this shows the simplicity of using Ch.

```
#include <math.h>
#include<fourbar.h>

Cfourbar fourbar;
class CPlot plot;
```

Program 2.7: Initialization for plotting of coupler curve for fourbar linkage (`fb_plot_initialization.ch`).

```

fourbar.setLinks(r1,r2,r3,r4, theta1);
fourbar.setCouplerPoint(rp, beta);
fourbar.setNumPoints(50);
fourbar.plotCouplerCurve(&plot,1);

```

Program 2.8: Plotting of coupler curve for fourbar linkage (fb_curve.ch).

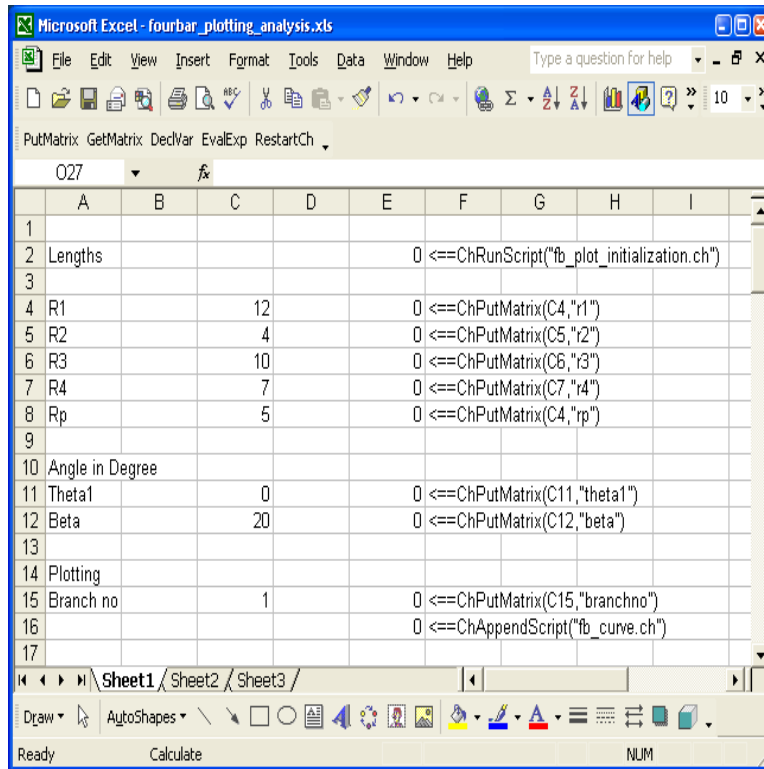


Figure 2.6: ChExcel spreadsheet for a fourbar linkage coupler curve. (four-bar-plotting_analysis.xls).

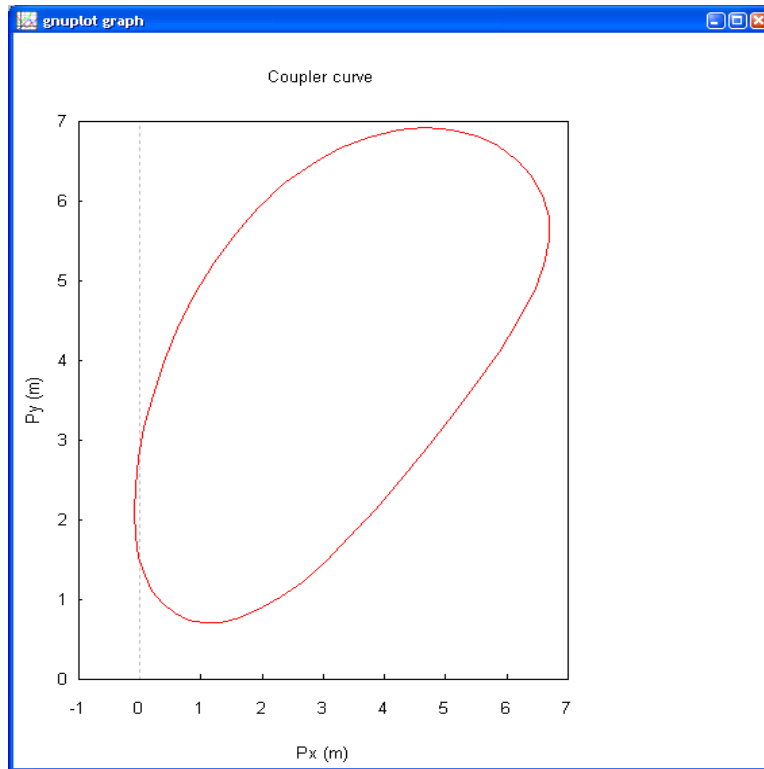


Figure 2.7: Plotting of a fourbar linkage coupler curve.

2.7 Angular Velocity Analysis

The Class `CFourbar` has functions for determining the angular velocity of the links in a fourbar linkage mechanism [6]. An Excel spreadsheet as shown in Figure 2.8 was designed for analysis of the angular velocity of all the links.

From the spreadsheet, the required input are obtained from the user. The comments in column A in spreadsheet are the representation of either input values or output values. The column C and D are used in getting input and displaying output. This column can be chosen by the user by changing the cell number in commands inserted in column E. For example, if the user wants to enter value of $R1$ in cell B4, the parameter of function `ChPutMatrix(C4,"r1")` in cell E4 is changed to `ChPutMatrix(B4,"r1")`. The column F indicates the function that has been written in the adjacent column. All values that has been given as input are sent to Ch script for processing and the output are then sent back again to the Excel sheet which are displayed at the required cells. The script `fb_vel_initialization.ch` performs the task of initialization of all the variables that are used in velocity analysis.

The script `is_link.ch` determines whether the proposed length of links can form a closed fourbar link or not. The output of this script is the display of a message box stating the possibility.

The various input values that have to be given are listed in the Excel spreadsheet. The known angle is selected from the drop down list from cell D16 and the link whose velocity is known from cell D19. The corresponding values are inserted in their respective cells as shown in Figure 2.8.

As the script files are executed in the required order, every input value is passed to Ch script for calculations.

```
theta[1] = theta1*M_PI/180; theta[angleno]=angle*M_PI/180;
omega_1[linkno]=velocity;
theta_2[angleno]=theta[angleno];

fourbar.angularPos(theta, theta_2, angleno);

omega_2[linkno] = omega_1[linkno];
if((fourbartype == FOURBAR_ROCKERCRANK)|| (fourbartype == FOURBAR_ROCKERROCKER)) {
    fourbar.angularVel(theta, omega_1, linkno);
    fourbar.angularVel(theta_2, omega_2, linkno);
} else {
    fourbar.angularVel(theta, omega_1, linkno);
    fourbar.angularVel(theta_2, omega_2, linkno);
}

for(i=1; i<=4; i++)
{
    omega_1deg[i] = omega_1[i]*180/M_PI;
    omega_2deg[i] = omega_2[i]*180/M_PI;
}
```

Program 2.9: Angular velocity analysis of a fourbar linkage (fb_angularVel.ch).

The fb_angularVel.ch script calculates the angular velocity of all the other links. The output is displayed in both radians per second and degrees per second. **ChGetMatrix(C24:C27,"omega_1")** in cell E24 displays the values stored in the array omega_1 from cells C24 through C27. This array has values in terms of rad per sec and variable omega_1deg in terms of deg per sec which are displayed in cells D24 through D27 using function **ChGetMatrix(D24:D27, "omega_1deg")** which is inducted in cell E26.

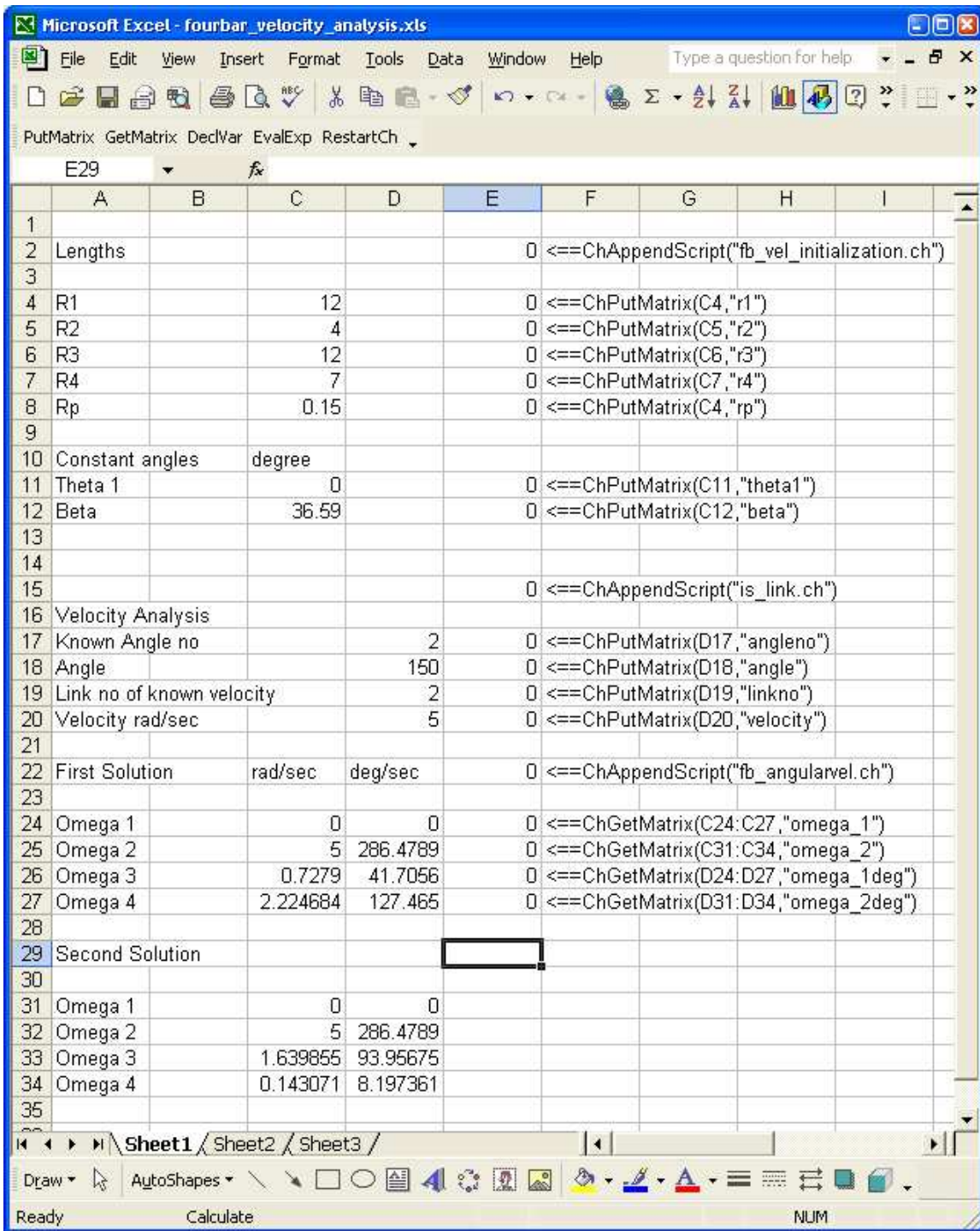


Figure 2.8: ChExcel spreadsheet for a fourbar linkage velocity analysis. (fourbar_velocity_analysis.xls).

2.8 Angular Acceleration Analysis

An Excel sheet shown in Figure 2.9 was designed for analysis of angular acceleration of fourbar mechanism. The design of this Excel sheet is similar to that of the velocity analysis Excel sheet. The initialization script file `fb_acc_initialization.ch` initializes all the required variables that would be used in the analysis of the angular acceleration of the links.

All the input values are to be inserted in the corresponding cells in the Excel spreadsheet as shown in Figure 2.9. The various input values that are required for this analysis are arranged in the spreadsheet. The script `fb_angularAccel.ch` when executed determines the angular acceleration of all the links which are then passed to Excel and placed at the required cells.

```
r[1] = r1; r[2] = r2; r[3] = r3; r[4] = r4;
theta[1] = theta1*M_PI/180;
theta[angleno] = angle*M_PI/180;
omega_1[linkno_vel] = velocity;
alpha_1[linkno_accel] = acceleration;

theta_2[angleno] = theta[angleno];
omega_2[linkno_vel] = omega_1[linkno_vel];
alpha_2[linkno_accel] = alpha_1[linkno_accel];

fourbar.angularPos(theta, theta_2, FOURBAR_LINK2);
if((fourbartype==FOURBAR_ROCKERROCKER)|| (fourbartype==FOURBAR_ROCKERCRANK))
{
    fourbar.angularVel(theta, omega_1, FOURBAR_LINK2);
    fourbar.angularAccel(theta, omega_1, alpha_1, FOURBAR_LINK2);
    fourbar.angularVel(theta_2, omega_2, FOURBAR_LINK2);
    fourbar.angularAccel(theta_2, omega_2, alpha_2, FOURBAR_LINK2);
} else {
    fourbar.angularVel(theta, omega_1, FOURBAR_LINK2);
    fourbar.angularAccel(theta, omega_1, alpha_1, FOURBAR_LINK2);
    fourbar.angularVel(theta_2, omega_2, FOURBAR_LINK2);
    fourbar.angularAccel(theta_2, omega_2, alpha_2, FOURBAR_LINK2);
}

for(i=1;i<=4;i++)
{
    alpha_1deg[i]=alpha_1[i]*180/M_PI;
    alpha_2deg[i]=alpha_2[i]*180/M_PI;
}
```

Program 2.10: Angular acceleration analysis of fourbar linkage. (`fb_angularAccel.ch`).

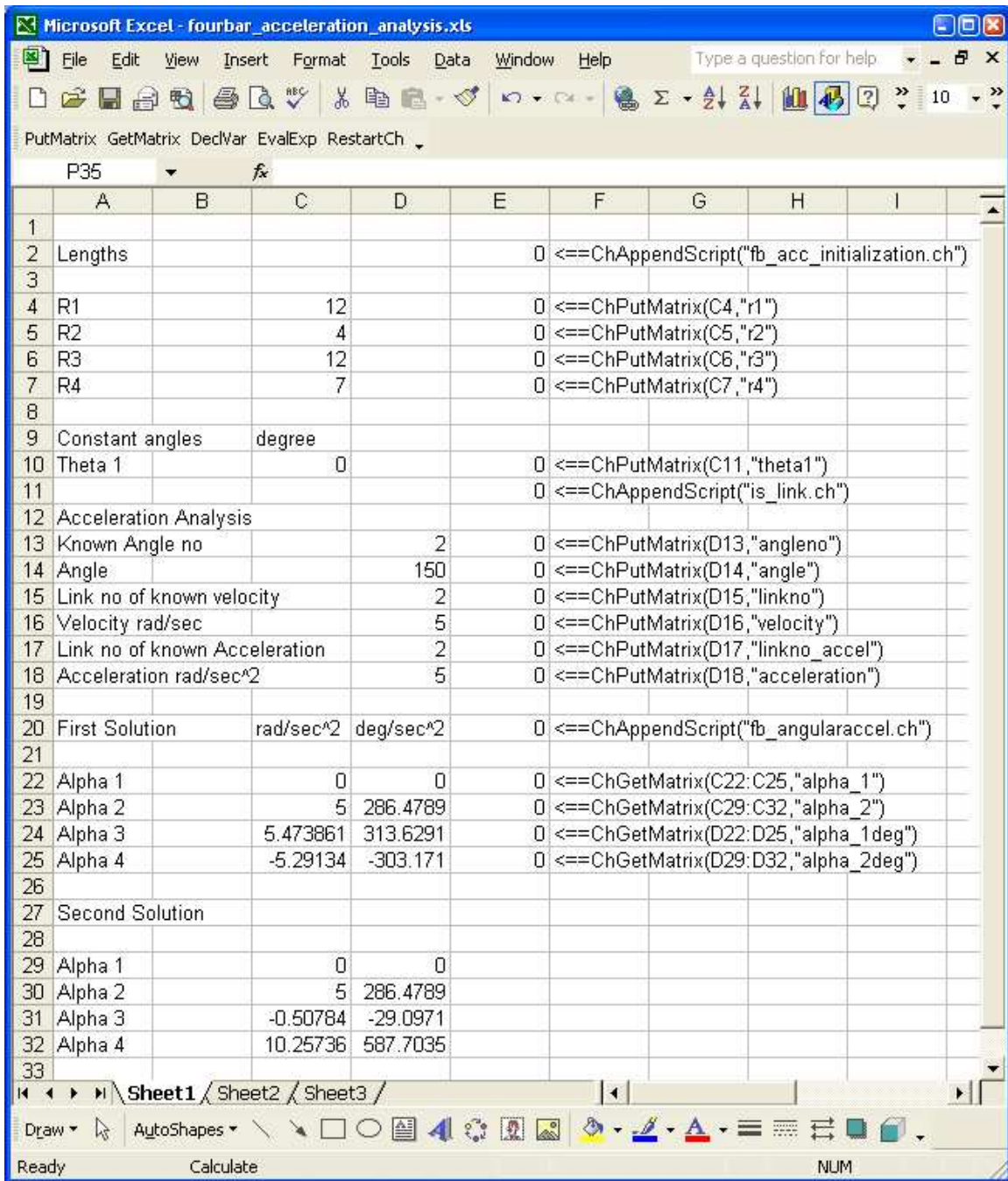


Figure 2.9: ChExcel spreadsheet for a fourbar linkage acceleration analysis. (fourbar_acceleration_analysis.xls).

2.9 Dynamic Analysis

The inertia-force analysis of the fourbar mechanism can be performed using the member function `forceTorque()` in the class `CFourbar`. This function can calculate the joint forces and the required input torque to achieve the desired motion of the fourbar linkage mechanism. The matrix

method has been used in formulation and analysis of dynamics of four bar mechanism. To simplify the programming burden, computational arrays in Ch are used for dynamic analysis.

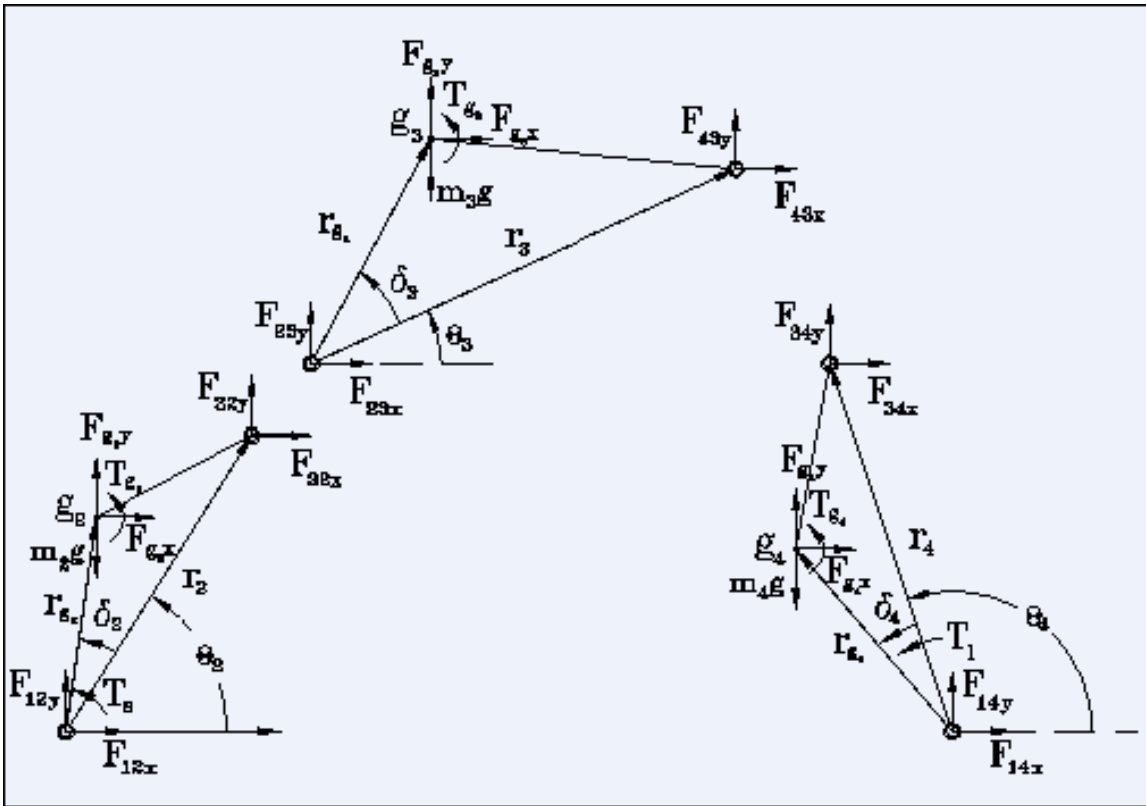


Figure 2.10: Free body diagram of fourbar linkage.

An Excel spreadsheet shown in Figure 2.11 was developed to analyze the dynamics of four-bar linkage mechanism. The spreadsheet has several variables and script files that have to be initialized and executed, respectively, are arranged in the order of the requirement. The script file `fb_force_initialization.ch` does all required declaration and initialization of all the variables that are used in the force analysis. All the link lengths and angles `theta1` and `beta` as shown in Figure 2.1 are initialized. All the necessary input data for this analysis are listed in the Excel sheet which also includes link's moment of inertia, angles that the corresponding links make with the ground. The other required parameters are the mass of links, magnitude of centre of gravity, deviation angles for center of gravity and the external load on joint of link1 and link4 as shown in Figure 2.10. The known angular velocity and angular acceleration of a link is given as input which can be selected from the drop down list.

When `fb_forceTorque.ch` is executed, the control goes to the Ch to perform the required calculations and the results are stored in form of array. It first determines the various angular accelerations of all the links and uses the member function `forceTorque()` to determine the reaction force in the joints. The output is then passed on to the Excel sheet where it is displayed in the required cell as shown in Figure 2.12.

```

m2 = mass2/g;
m3 = mass3/g;
m4 = mass4/g;
beta = beta*M_PI/180;
t1 = load;

theta_1[1] = 0;
theta_1[angleno]=angle*M_PI/180;
theta_2[1] = 0;
theta_2[angleno]=angle*M_PI/180;
omega_1[linkno_vel] = velocity;
alpha_1[linkno_accel] = acceleration;
omega_2[linkno_vel] = velocity;
alpha_2[linkno_accel] = acceleration;

fourbar.setLinks(r1, r2, r3, r4, thetal);
fourbar.setCouplerPoint(rp, beta);
fourbar.setGravityCenter(rg2, rg3, rg4, delta2, delta3, delta4);
fourbar.setInertia(ig2, ig3, ig4);
fourbar.setMass(m2, m3, m4);

fourbar.angularPos(theta_1, theta_2, FOURBAR_LINK2);
fourbar.angularVel(theta_1, omega_1, FOURBAR_LINK2);
fourbar.angularAccel(theta_1, omega_1, alpha_1, FOURBAR_LINK2);
fourbar.forceTorque(theta_1, omega_1, alpha_1, t1, X_1);

fourbar.angularVel(theta_2, omega_2, FOURBAR_LINK2);
fourbar.angularAccel(theta_2, omega_2, alpha_2, FOURBAR_LINK2);
fourbar.forceTorque(theta_2, omega_2, alpha_2, t1, X_2);

```

Program 2.11: Force analysis of fourbar linkage (fb_forceTorque.ch).

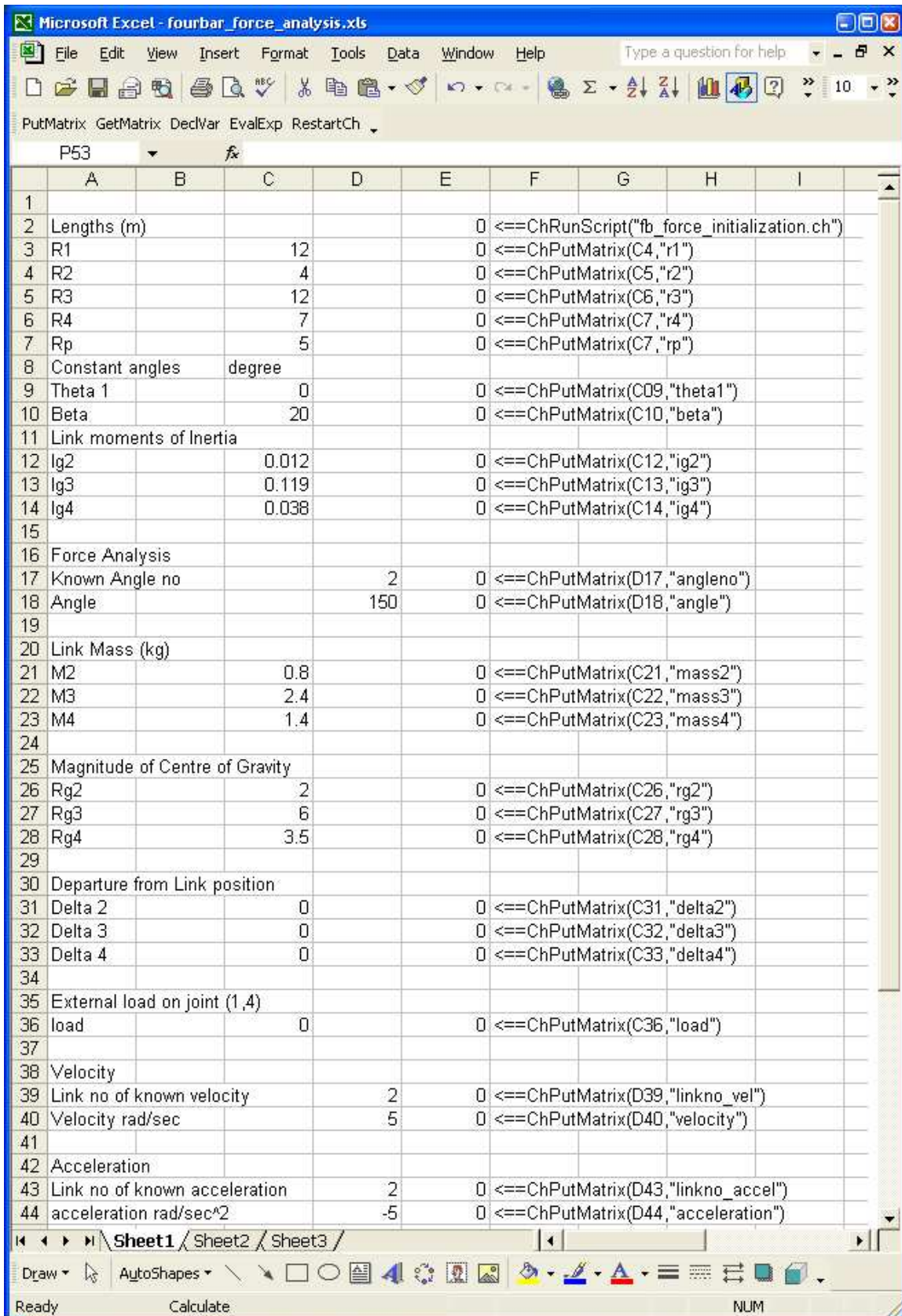


Figure 2.11: ChExcel spreadsheet for fourbar dynamic analysis. (fourbar_force_analysis.xls).

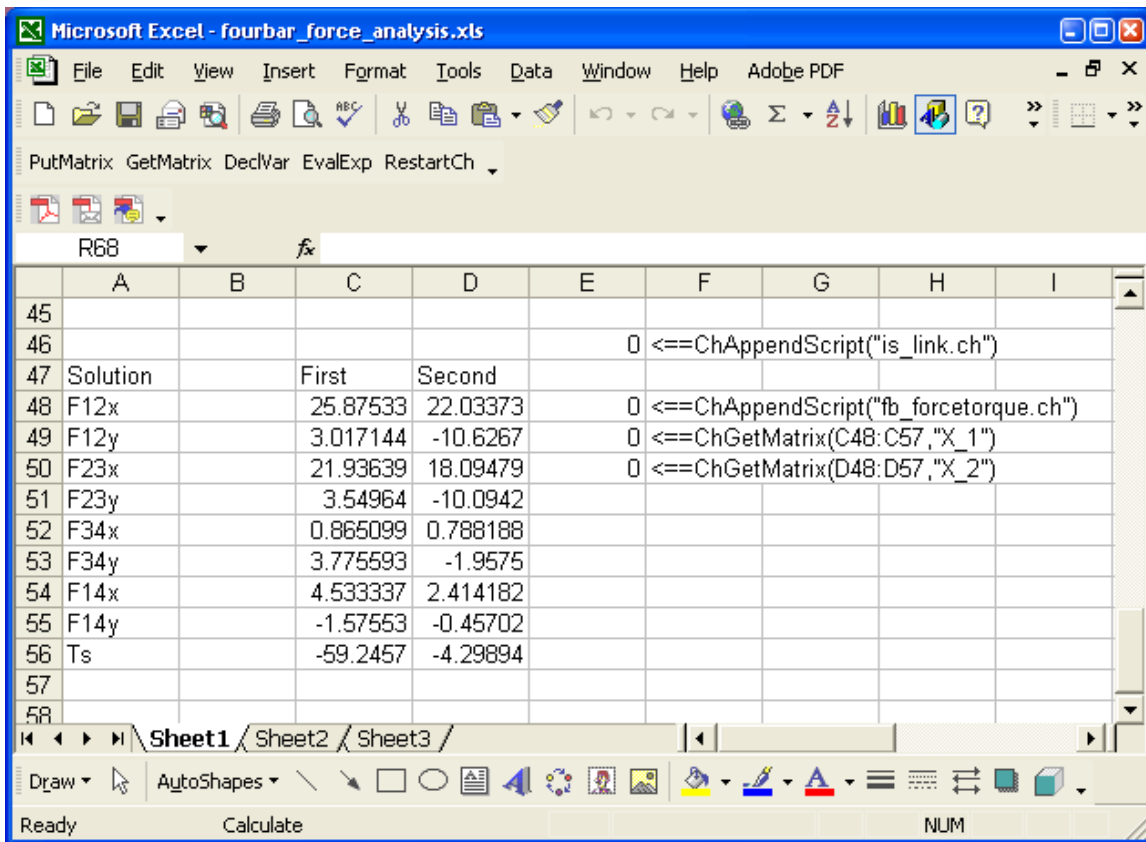


Figure 2.12: ChExcel spreadsheet for fourbar dynamic analysis. (fourbar_force_analysis.xls) (Contd.).

Chapter 3

Analysis of Other Mechanism Using Ch and ChExcel

The other mechanism classes of the Ch mechanism Toolkit [6] includes

- CGearedFivebar - Geared Fivebar mechanism
- CCrankSlider - Crank slider mechanism
- CFourbarSlider - Fourbar Slider Mechanism
- CStevSixbarI - Stephenson I mechanism
- CStevSixbarIII - Stephenson I mechanism
- CWattSixbarI - Watt I mechanism
- CWattSixbarII - Watt II mechanism

The design and implementation of other mechanism are similar to those of the fourbar mechanism that has been presented in the previous chapter. An Excel spreadsheet has been developed for each mechanism mentioned above for specific requirements. These Excel sheets are described in this chapter.

3.1 Position Analysis of Geared Fivebar Mechanism

The Geared fivebar linkage as shown in Figure 3.1 is considered and its position analysis is performed. The class CGearedFivebar was developed with various member functions using which the problems pertaining to its analysis can be solved. Similar to the fourbar mechanism described in the previous chapter, geared fivebar mechanism is solved using a set of equations to determine the kinematic properties of each link as well as a coupler point. An Excel sheet in Figure 3.2 was developed to determine the angular position of the geared fivebar linkage.

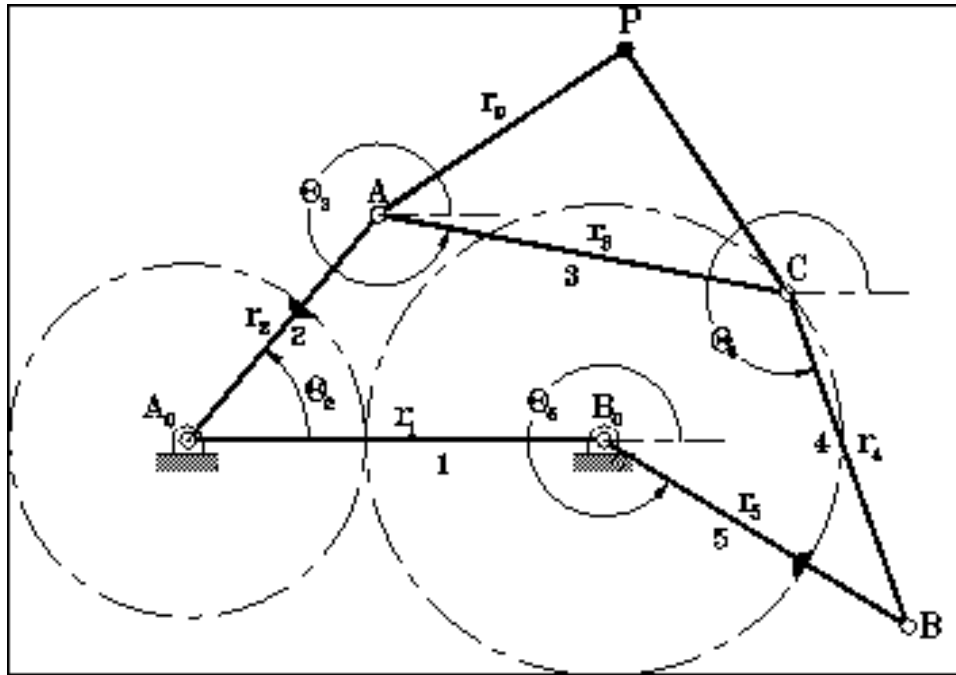


Figure 3.1: Geared fivebar mechanism.

The spreadsheet in Figure 3.2 has been designed to obtain the required input in predetermined order. The user may have to go in the same order to input the data and execute the commands in the order they are listed. The `cg_initialization.ch` initializes all the variables that are used in determining the angular positions of all links in a geared fivebar linkage shown in Figure 3.1. The various equations are solved internally using `complexsolve()` function. The various inputs are obtained through the Excel sheet. The `cg_angularPos.ch` uses the `CGearedFivebar` class and its member functions to do the required calculations and the results are then stored in a `Ch` variable. This `Ch` variable is directly called from Excel spreadsheet and displayed in the required cells as shown in Figure 3.2.

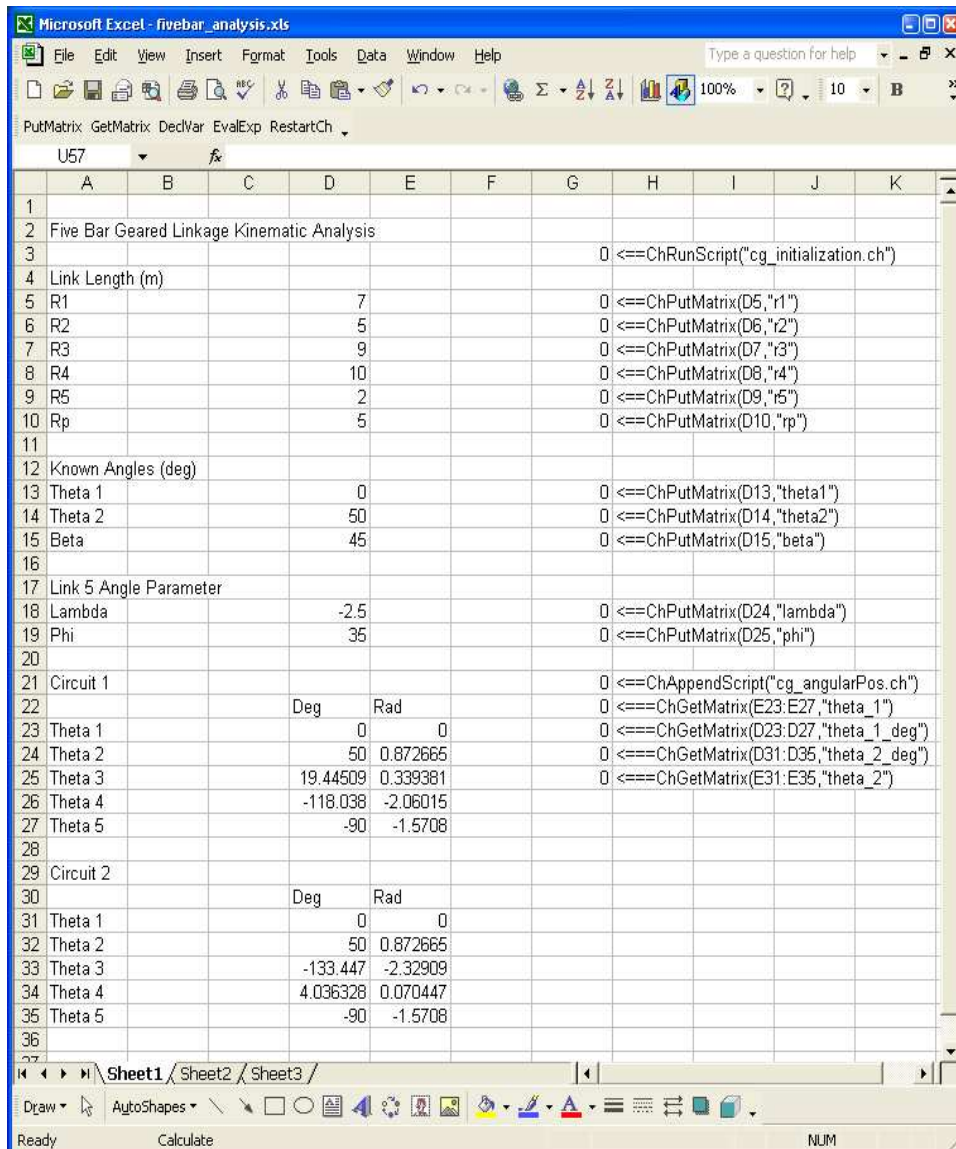


Figure 3.2: ChExcel spreadsheet for position analysis of a geared fivebar mechanism (five-bar_analysis.xls).

3.2 Slider Position Analysis of Crank Slider Mechanism

For a crank slider as shown in Figure 3.3, a spreadsheet was developed to analyze the slider position at a given instant. The class `CCrankSlider` in the mechanism toolkit has various member functions which analyze and finds the position, velocity and acceleration of slider. Furthermore there are member functions which determine the angular position, velocity and acceleration of the links at a given instant.

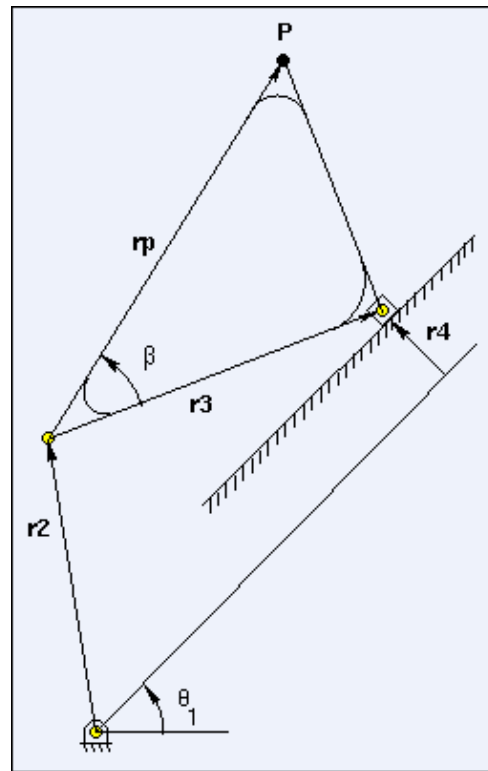


Figure 3.3: Crank Slider Mechanism.

Figure 3.4 shows the snap shot of Excel sheet that was developed to determine the position of the slider of a crank slider mechanism. The script file `cs_sp_initialization.ch` initializes the variables used in slider position analysis. The various input are entered in the corresponding cells, like the length of link, angles θ_1 , θ_2 and β as shown in Figure 3.3. Script file `cs_sliderPos.ch` determines the current position of slider for the given input data. It was also found that there is possibility of two circuit being formed using the given input data and the position of slider for both circuits is identified and displayed in the required cells as shown in Figure 3.4.

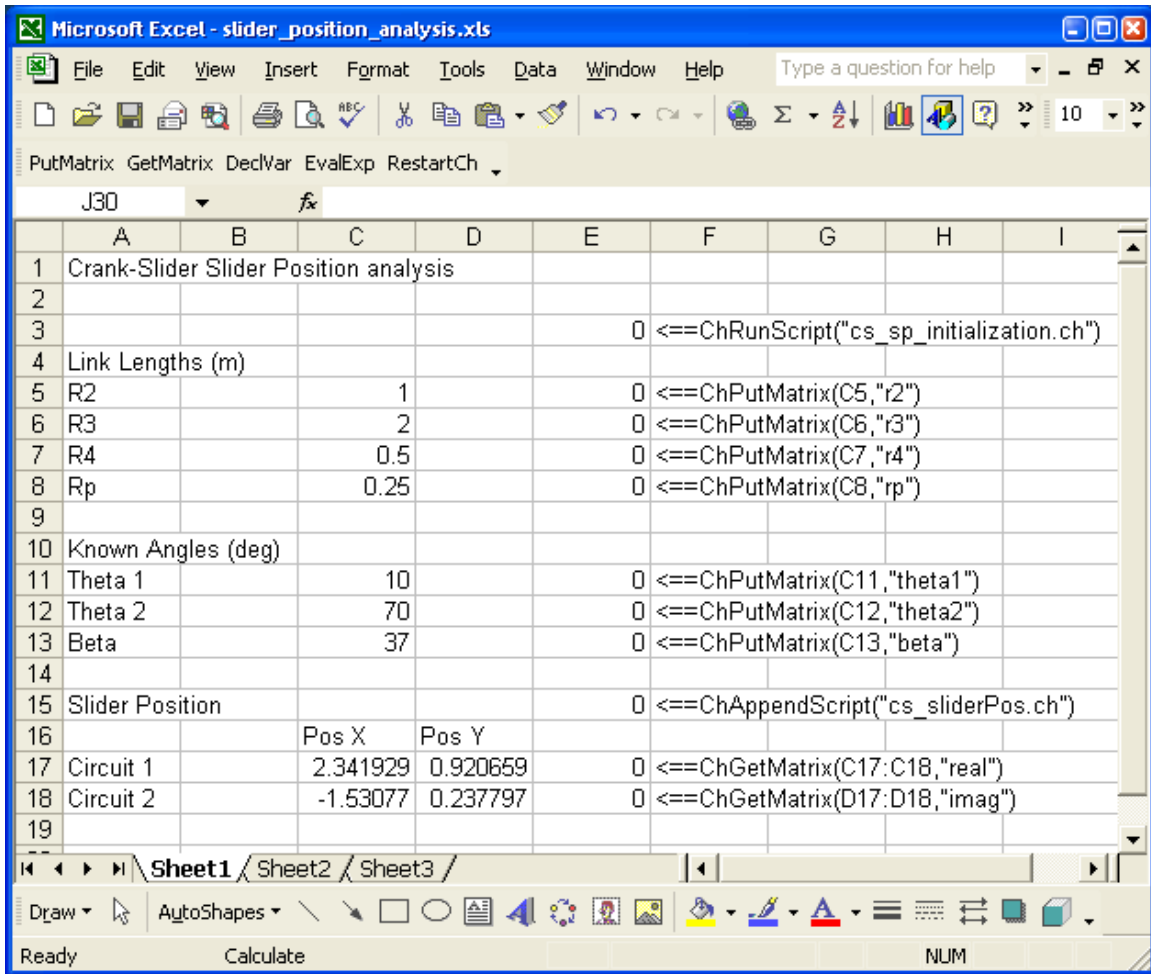


Figure 3.4: ChExcel spreadsheet for the analysis of slider position of crank slider mechanism (slider_position_analysis.xls).

3.3 Position Analysis of Multi-Loop Six Bar Linkages

Among the various multi-loop six bar linkage fourbar slider, Watt six-bar(I) linkage, Watt six-bar(II) linkage, Stephenson six bar(I) Linkage and Stephenson six-bar(III) linkage were considered. The Ch mechanism toolkit contains classes for these linkages which have member functions to determine the angular positions, angular velocities and angular accelerations of a link at any given instant. The following sections explain more on analyzing these mechanisms from Excel.

3.3.1 Position Analysis of Fourbar Slider Mechanism

For a fourbar-slider mechanism shown in Figure 3.5, a spreadsheet as shown in Figure 3.6 and 3.7 was developed to analyze the angular position of various links in the mechanism at a given instant. The class CFourbarSlider was developed with member functions to determine the angular position of the links, their angular velocities, angular accelerations and also its coupler

point positions, velocities and accelerations. This spread sheet (Figure 3.6, 3.7) analyze the angular positions of links.

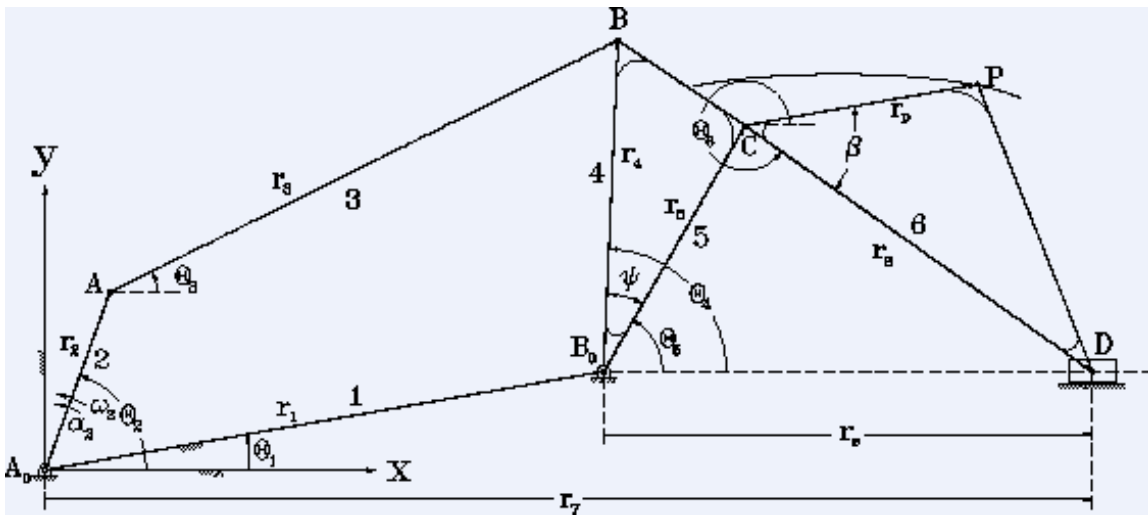


Figure 3.5: Fourbar slider mechanism position analysis.

The various input data such as the length of links and angles made by the link to the ground, β and ψ , the angle between the link 4 and link5 as shown in Figure 3.5 are given as input. The variables used in position analysis of crank slider mechanism are initialized in `cfs_initialization.ch`. All the input values are stored in variables that are shown in the corresponding **ChPutMatrix()** function argument.

The script `cfs_angularPos.ch`, when executed determines the angular positions of various links in the mechanism and stores them in an array. This array is displayed in Excel spreadsheet at the specified cells using the function **ChGetMatrix()**.

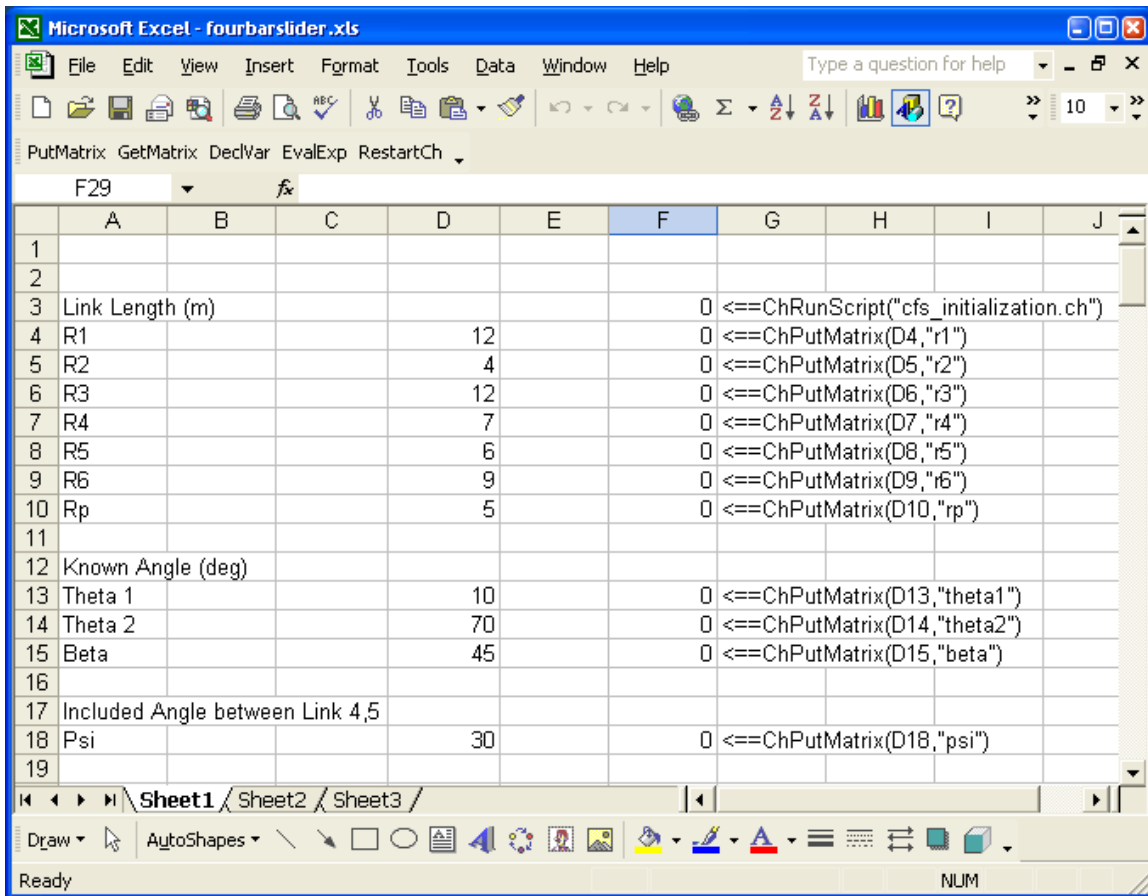


Figure 3.6: ChExcel spreadsheet for position analysis of a fourbar slider mechanism (fourbarslider.xls).

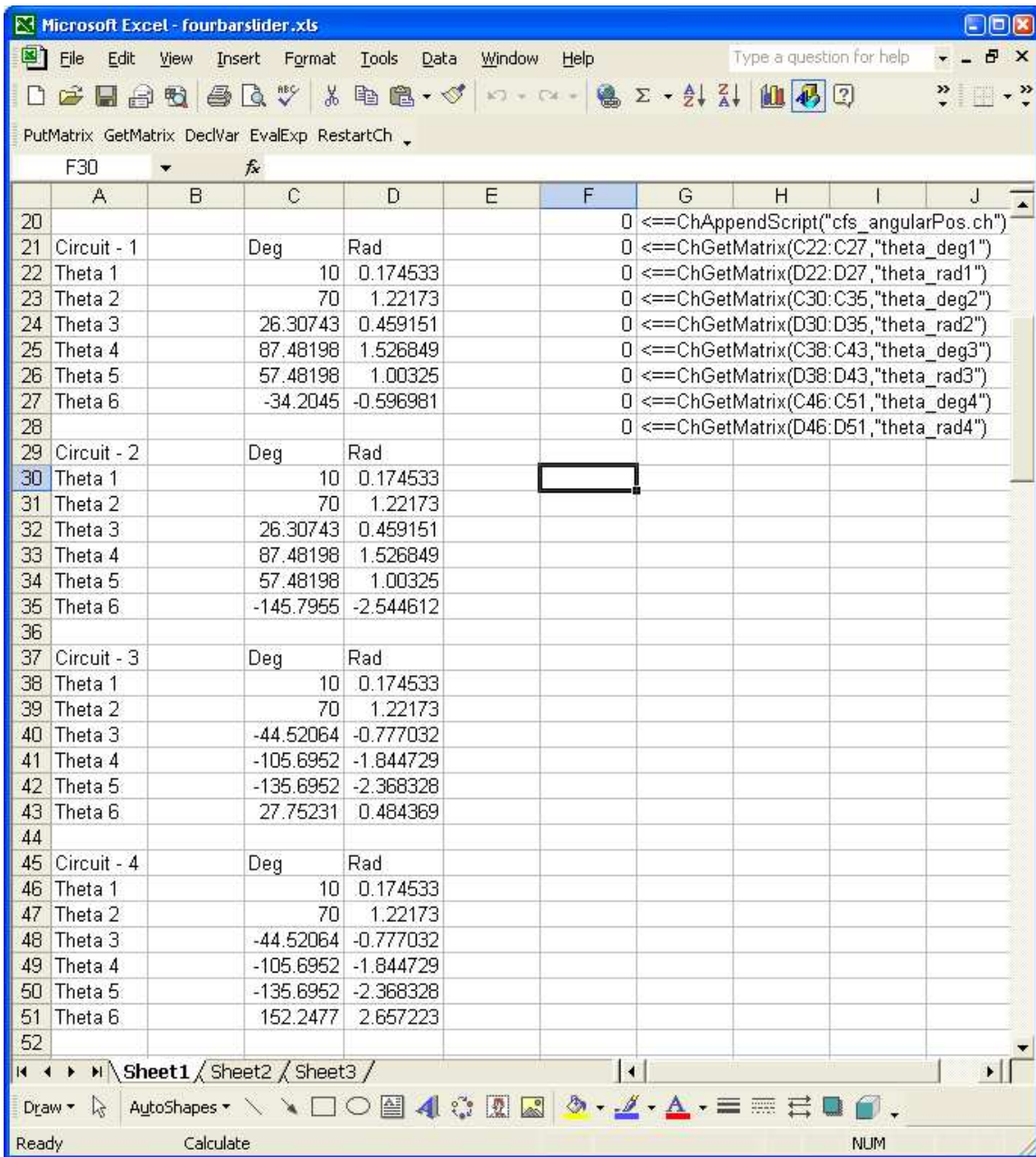


Figure 3.7: ChExcel spreadsheet for position analysis of a fourbar slider mechanism (fourbarslider.xls)(contd.).

3.3.2 Position Analysis of Watt Six-Bar(I) linkage

A Watt (I) Six bar linkage is assembled from two four bar mechanisms as shown in Figure 3.8. A spreadsheet was designed and developed to analyze the angular position of various links at a given instant. Figure 3.9 shows the input cells of the spreadsheet where the values are entered by user. The corresponding cells in column F are executed to store the values in the variables. The script `ws1_initialization.ch` initializes all variables used in this analysis in Ch. The input values

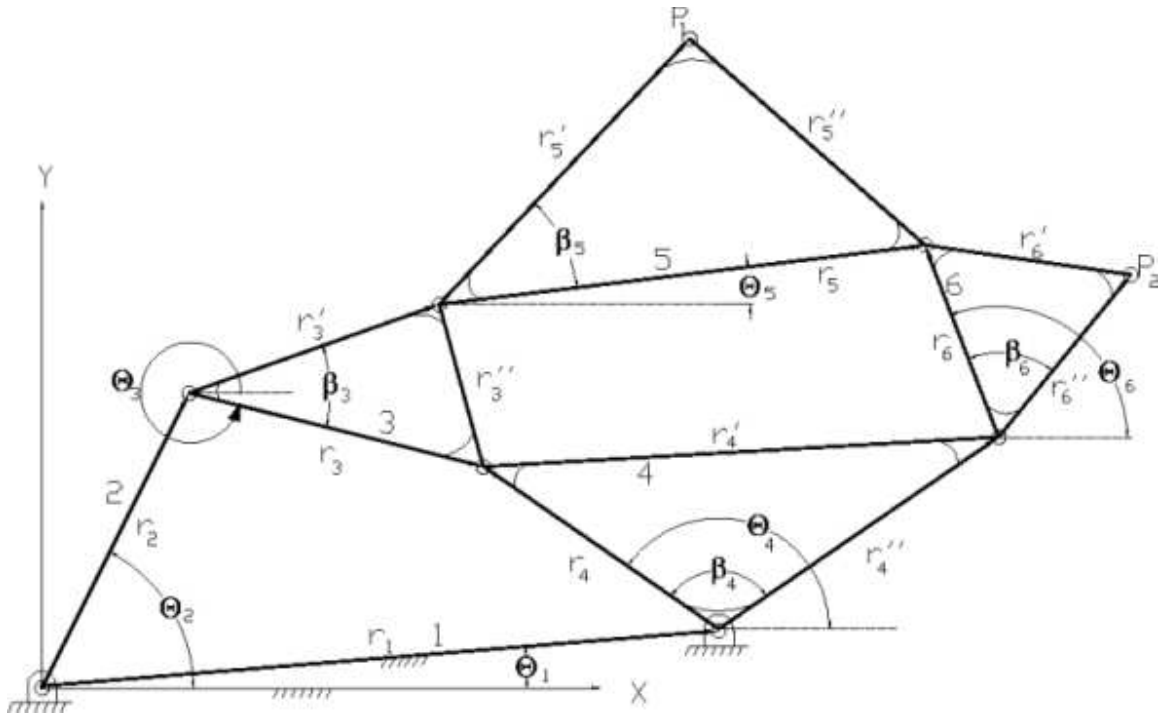


Figure 3.8: Watt (I) Six-Bar Mechanism.

are the length of links as shown in Figure 3.8, where R_{p3} indicates r'_3 in the figure, R_{pp4} indicates r''_4 , R_{p5} indicates r'_5 and R_{pp6} indicates r''_6 in the Figure 3.8. The other values required are listed in the Excel sheet.

The script `ws1_angularPos.ch` finds the angular positions of various links in the mechanism. It also determines the position in all the possible branches that can be formed with the given link specification. On analysis it was found to have four different branches for the given specification and all four different set of values are determined and returned to Excel spreadsheet as shown in Figure 3.10.

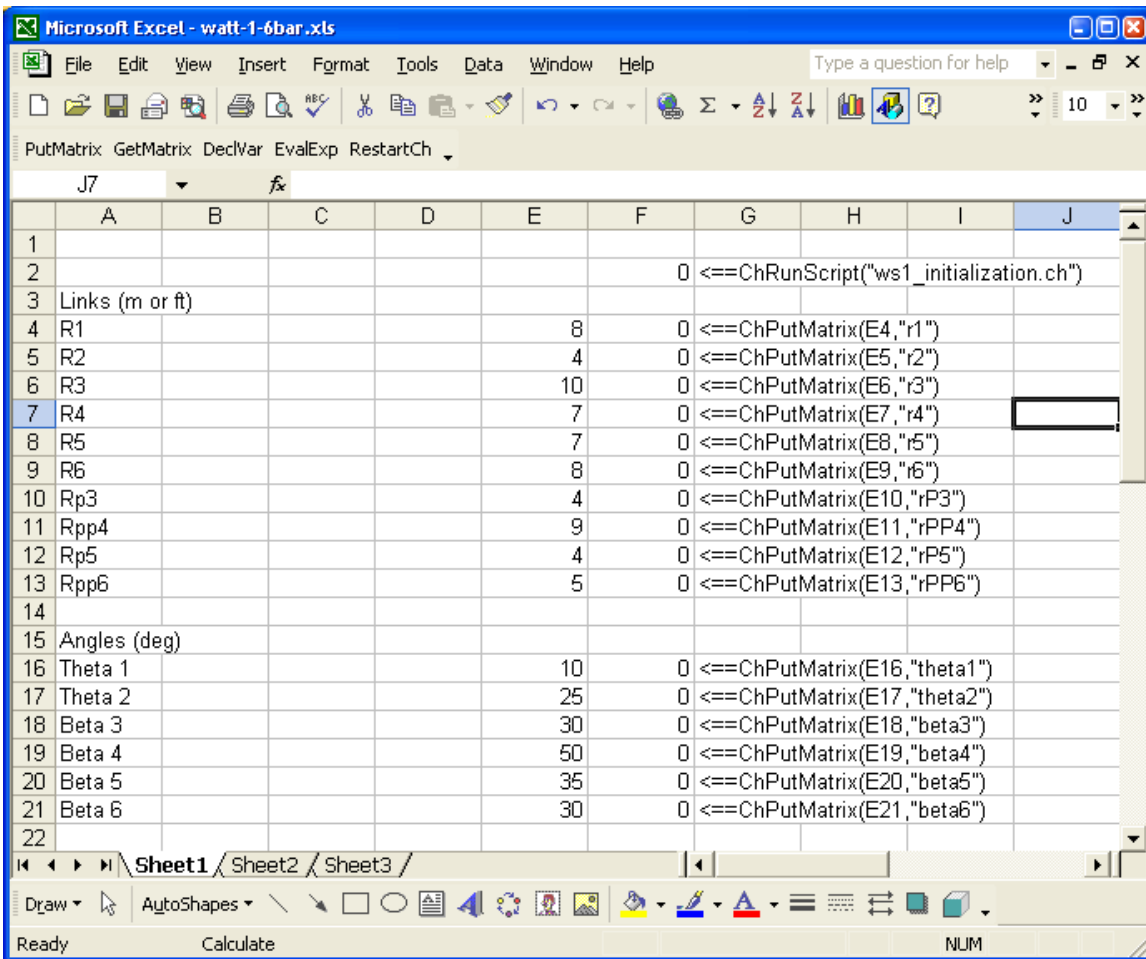


Figure 3.9: ChExcel spreadsheet for position analysis of Watt (I) six-bar mechanism(watt-1-6bar.xls).

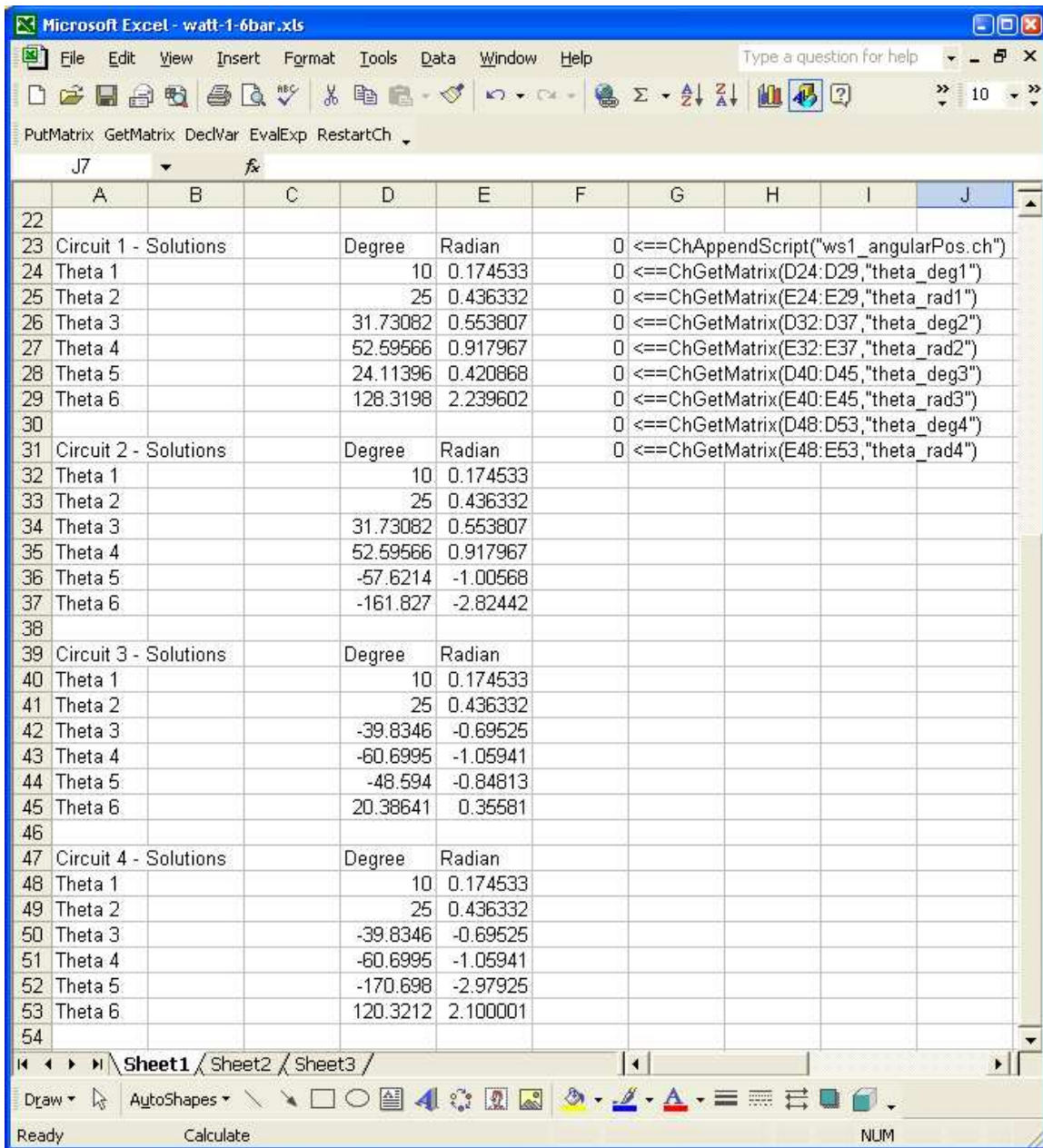


Figure 3.10: ChExcel spreadsheet for position analysis of Watt (I) Six-Bar Mechanism (contd.).

3.3.3 Position Analysis of Watt Six-bar (II) Linkage

A Watt(II) sixbar linkage is formed from two fourbar mechanisms where the output link of the first fourbar is the input link of the second fourbar linkage. For the Watt(II) linkage in Figure 3.11, there is also a coupler point located on the floating link of the second fourbar linkage. An Excel spreadsheet in Figure 3.12 was designed and developed to determine the angular positions of various links at a given instant. The instant is defined by the angles namely theta1, theta2, theta5, beta and psi as shown in Figure 3.11. In Figures 3.12 and 3.13, column A represents the variable which is either input or output, column E has values that are entered by the user. Cells in column

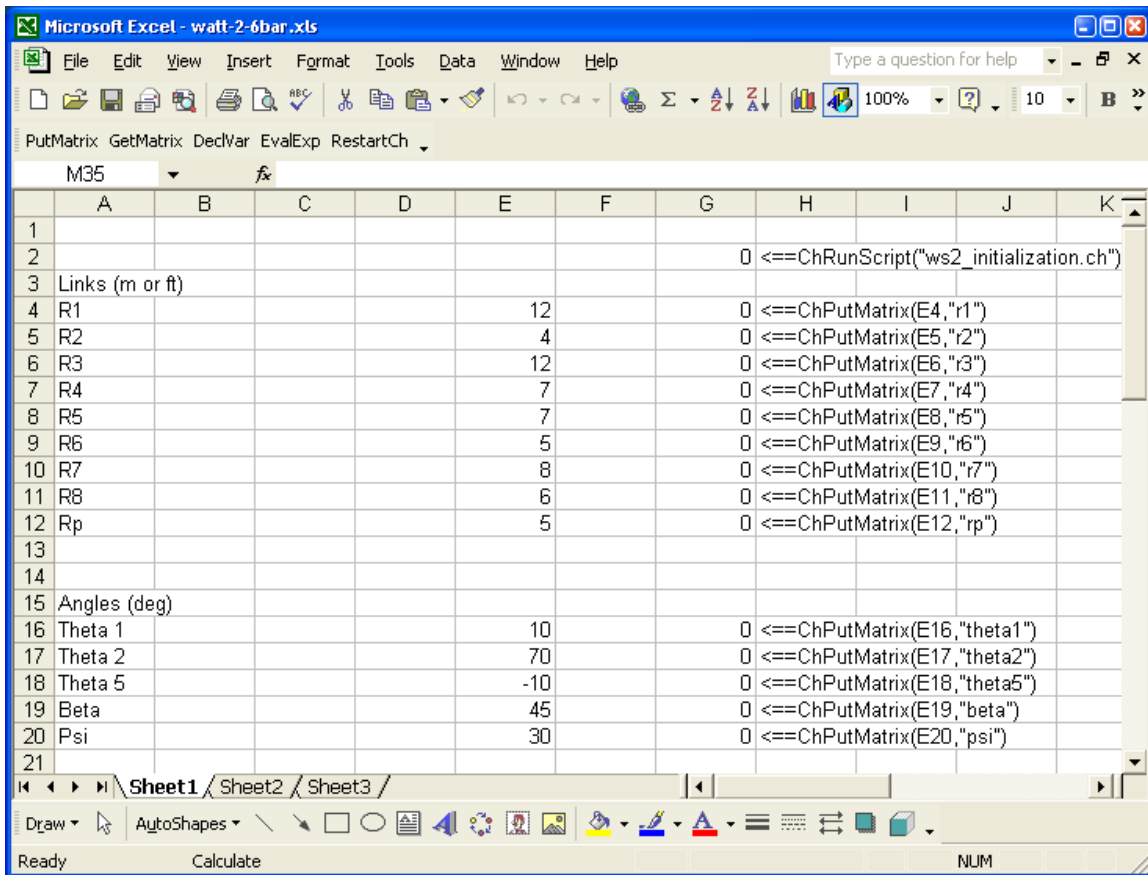


Figure 3.12: ChExcel spreadsheet for position analysis of watt (II) six-bar mechanism(watt-2-6bar.xls).

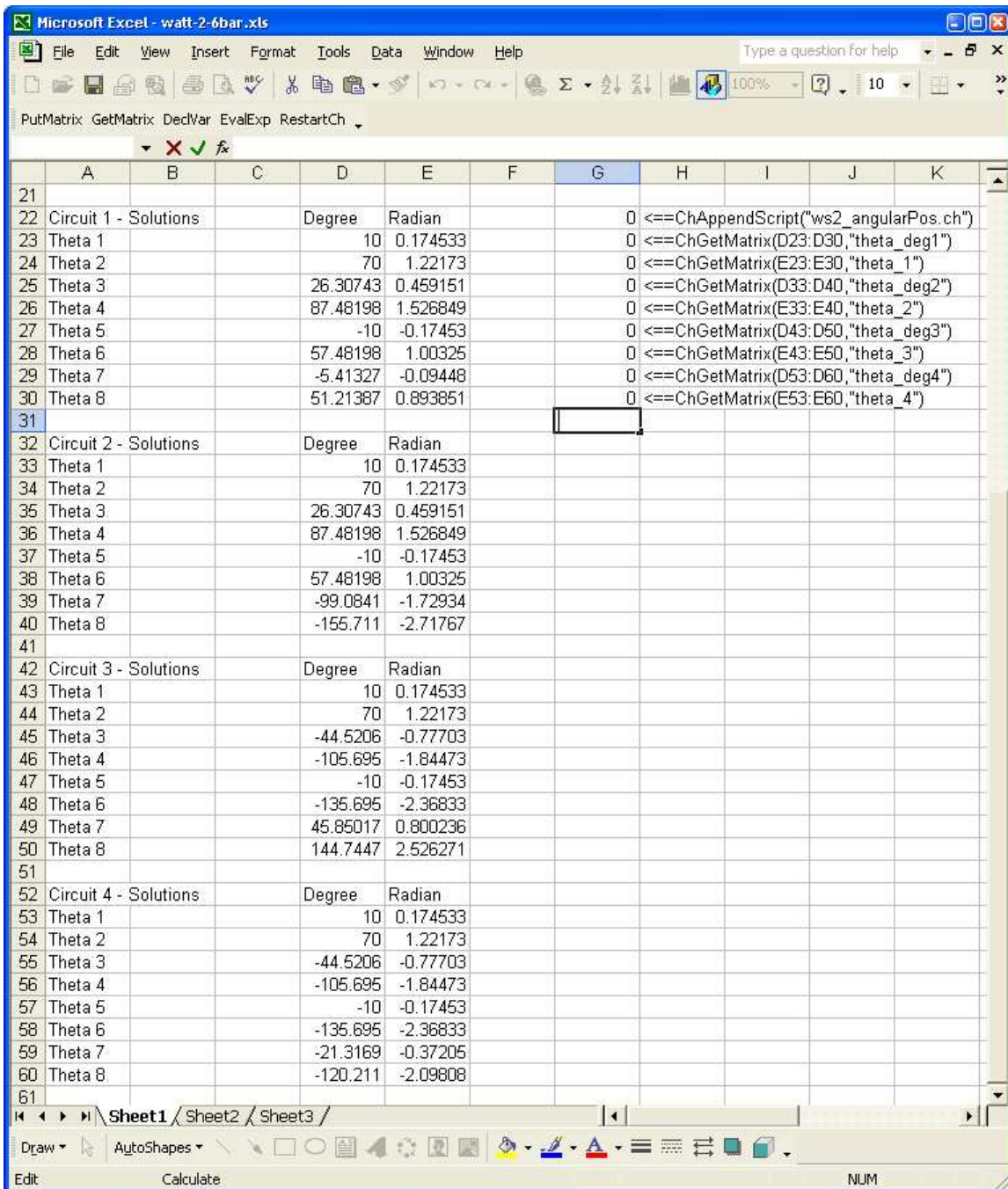


Figure 3.13: ChExcel spreadsheet for position analysis of watt (II) six-bar mechanism (watt-2-6bar.xls) (contd.).

3.3.4 Stephenson Six-bar (I) Linkage

A Stephenson (I) mechanism is composed of a fourbar linkage and a restricted moving fivebar linkage as shown in Figure 3.14. The rigid body input and output links of fourbar linkage moves the links of the fivebar linkage, thereby a relation is created between the input and output of

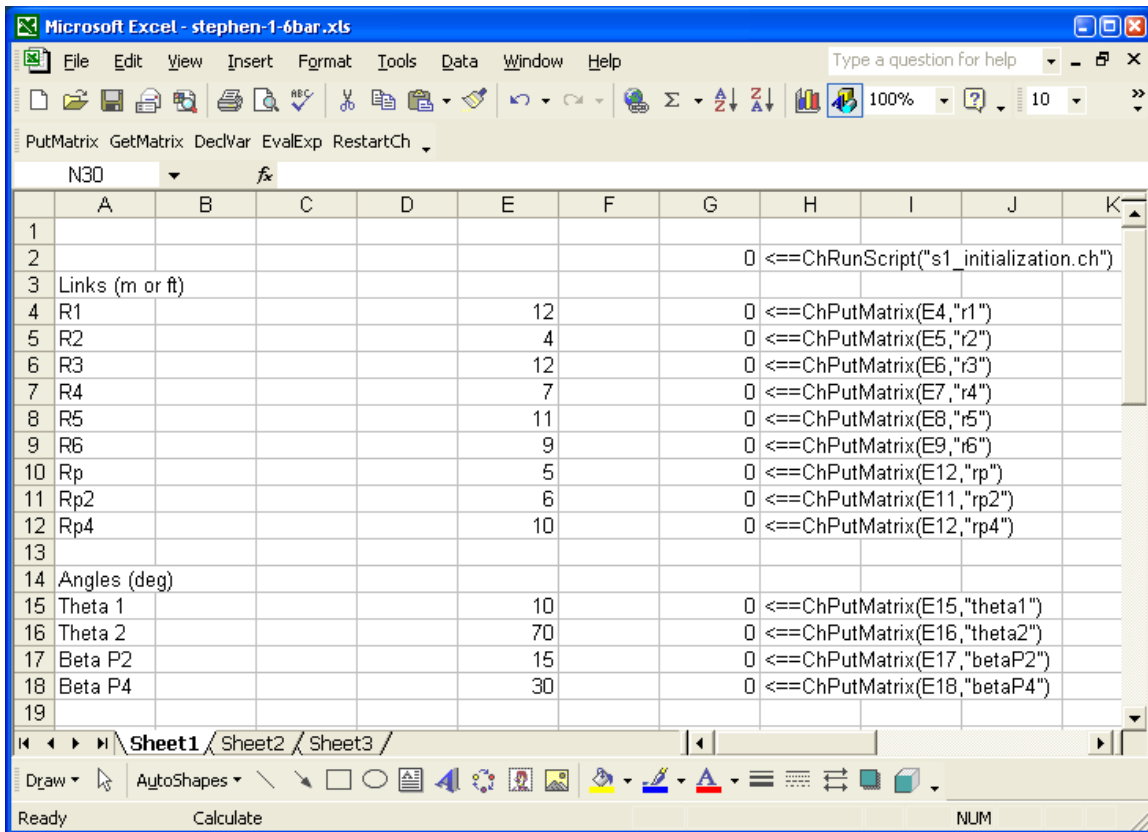


Figure 3.15: ChExcel spreadsheet for position analysis of Stephenson (I) six-bar mechanism (stephen-1-6bar.xls).

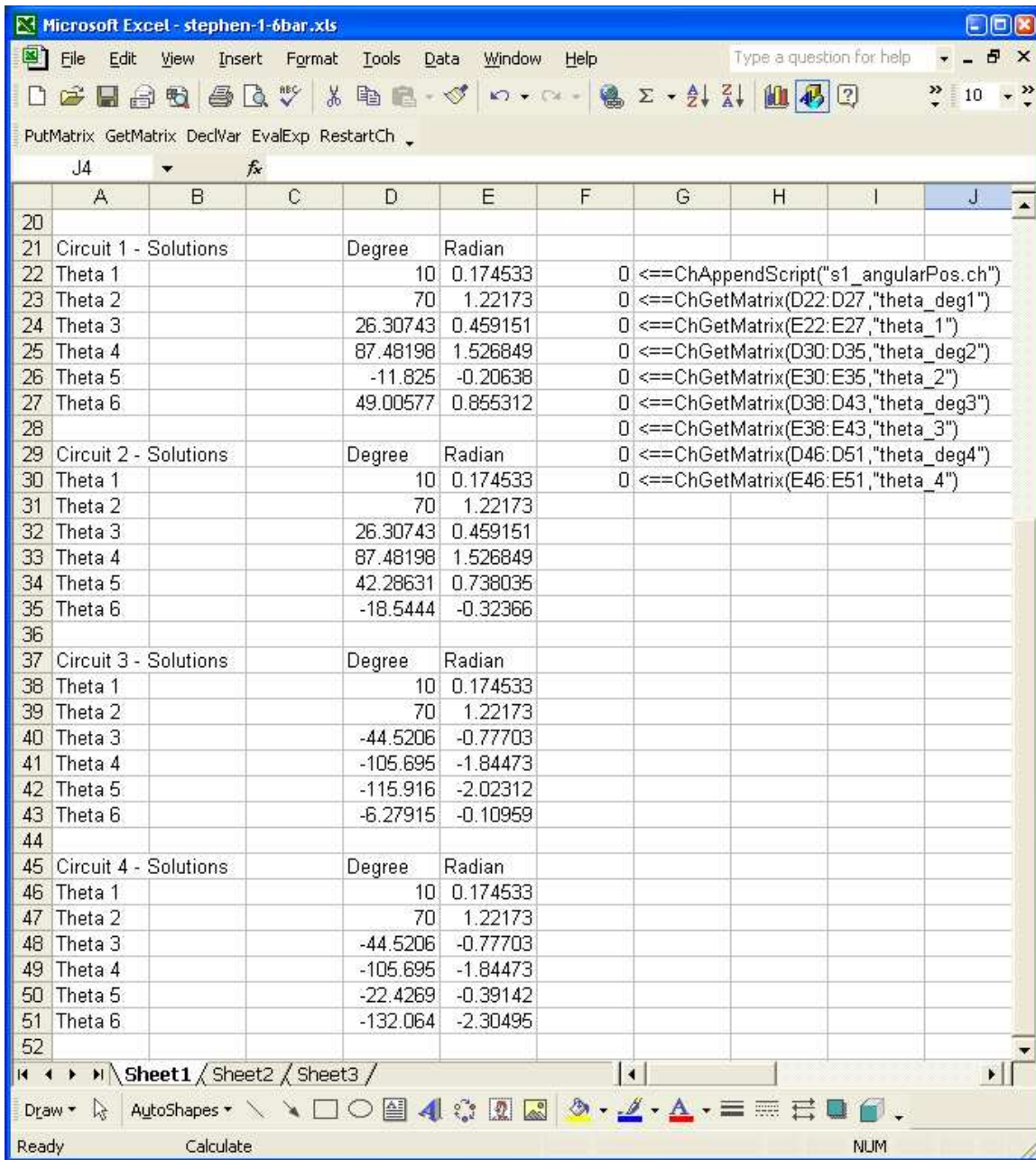


Figure 3.16: ChExcel spreadsheet for position analysis of Stephenson (I) six-bar mechanism (stephen-1-6bar.xls)(contd.).

3.3.5 Stephenson Six-bar (III) Linkage

A Stephenson (III) mechanism is composed of a normal fourbar linkage and a restricted moving fourbar as shown in Figure 3.17. The output link of the first fourbar linkage becomes the input of the second fourbar linkage. The limited motion of the fourth link results in the restrictive motion of the second fourbar. In order to determine the angular position of various links, an Excel spreadsheet was developed as shown in Figures 3.18 and 3.19.

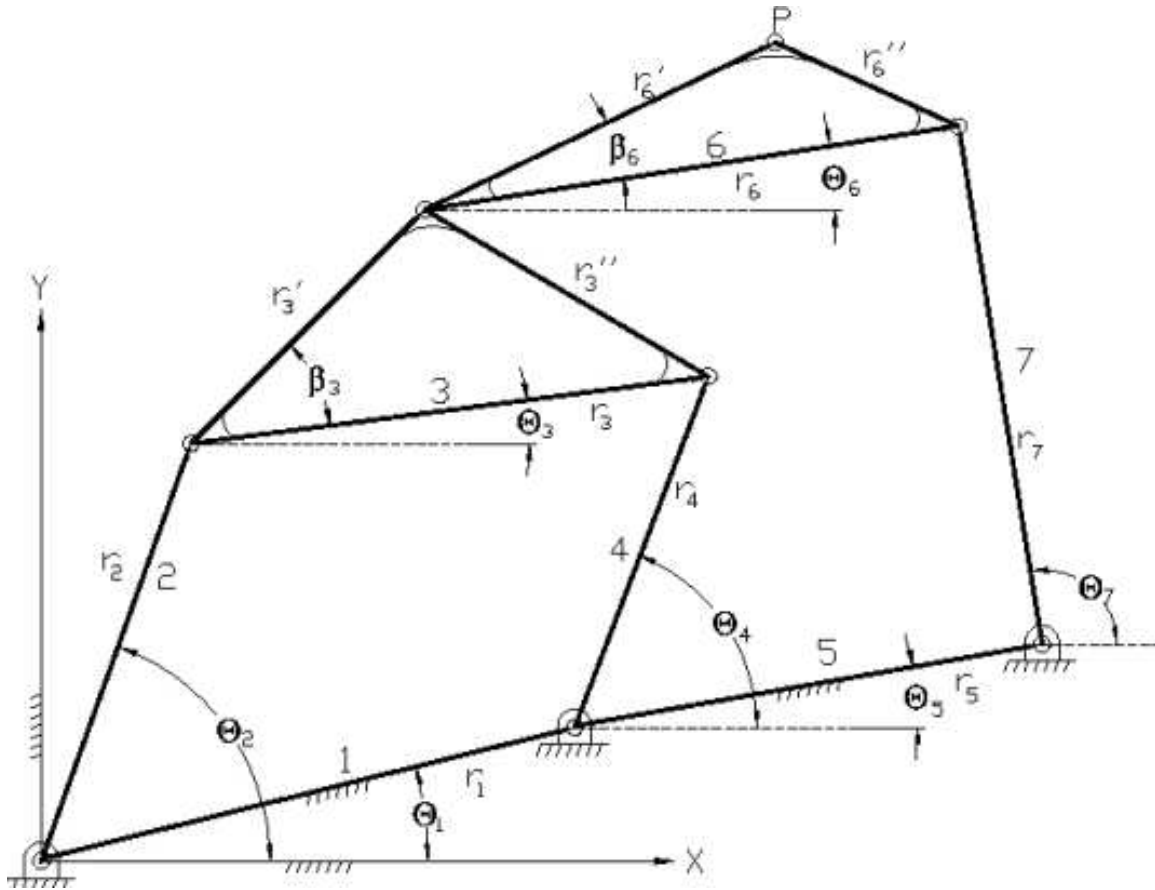


Figure 3.17: Stephenson (III) six-bar mechanism.

The input values are entered in cells of column E which are read by the function inserted in cells of column G. The script file `s3_initialization.ch` when executed initializes all the variables used in this analysis. All input values as shown in Figure 3.18 are stored in the variables using ChExcel function **ChPutMatrix()**. On executing the script file `s3_angularPos.ch` using ChExcel function **ChAppendScript()**, the angular position of various links are determined and are displayed in the specified cells using ChExcel function **ChGetMatrix**. For the given set of input values it was found that there is a possibility of four different circuits. The angular positions of all links for all the circuits are determined and stored in a Ch variable. This variable is called from Excel and its contents are displayed in spreadsheet as shown in Figure 3.19.

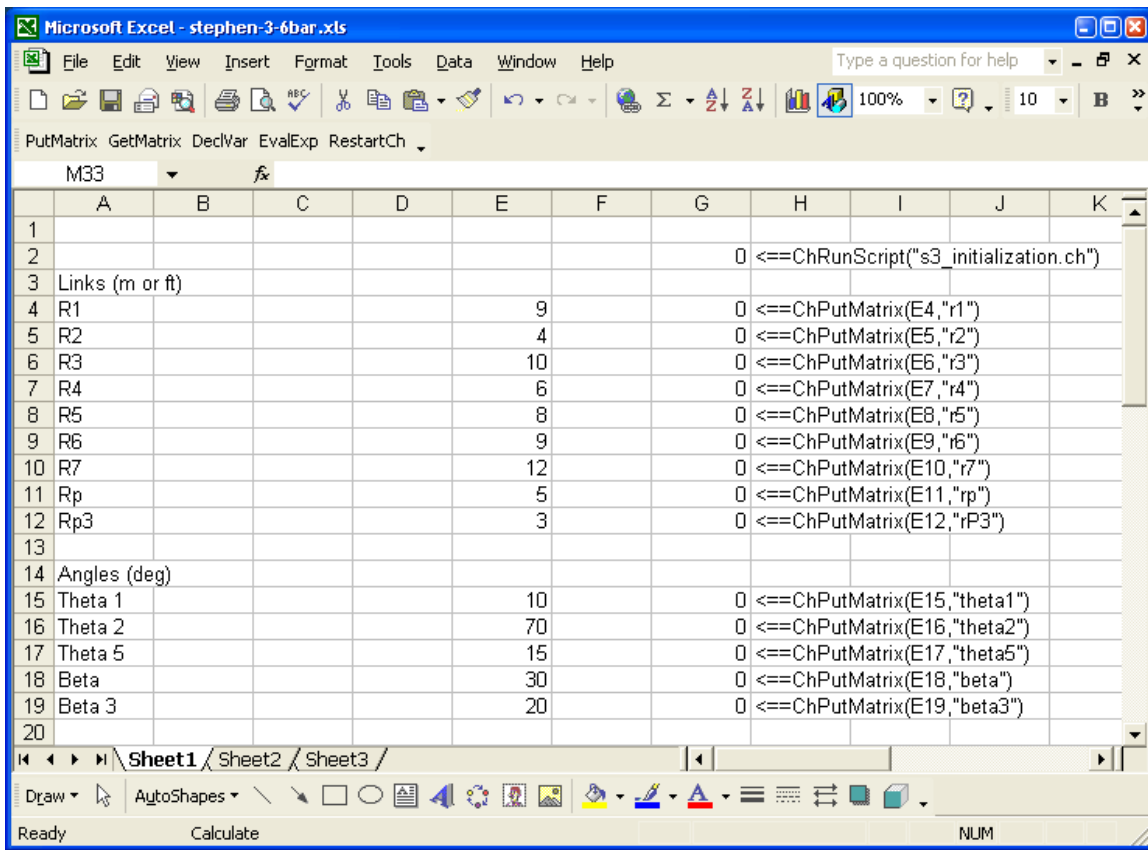


Figure 3.18: ChExcel spreadsheet for position analysis of Stephenson (III) six-bar mechanism (stephen-3-6bar.xls).

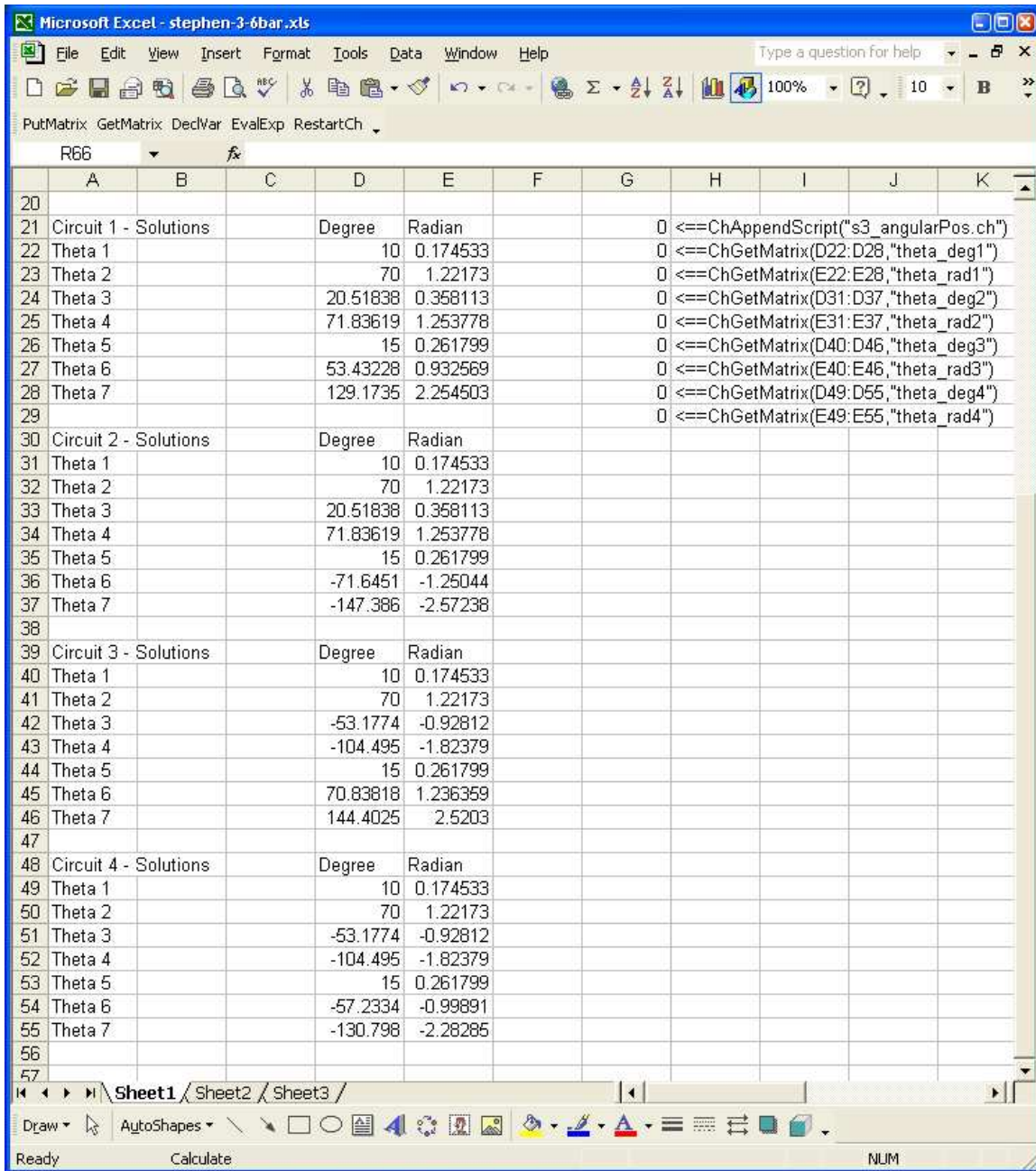


Figure 3.19: ChExcel spreadsheet for position analysis of Stephenson (III) six-bar mechanism (stephen-3-6bar.xls)(contd.).

Chapter 4

Conclusions

This research work is mainly focused on applying Ch to solve engineering problems. Ch Excel has been used to solve mechanism analysis and design problems.

In this work, ChExcel was used for analysis of various mechanisms. It highlights the usage of computational power of Ch from Excel which acts as the front end for GUI whereas Ch at the back-end performs all the required computations. Because it is user friendly, Excel is commonly used software by engineers for various computational tasks. Ch scripts executed in the back end follow the well established programming paradigm in C. This provides a great flexibility for developers in developing software modules. By using Ch, the developers can take advantage of other software modules, toolkits and packages in Ch. For example, in this project, QAnimate was used in generating animation for mechanisms. Instead of developing a new software module, the available components in Ch can be used for variety of purposes.

Bibliography

- [1] Harry H. Cheng. Scientific Computing in the Ch Programming Language . *Scientific Programming*, 2(3):49–75, 1993.
- [2] Harry H. Cheng. *C99 for Engineers and Scientists*. IEL , UCD, 2004. See also URL <http://www.softintegration.com>.
- [3] Heron Technologies. URL <http://www.heron-technologies.com>.
- [4] Parametric Technology Corporation,. URL <http://www.ptc.com>.
- [5] Artas,. URL <http://www.artas.nl>.
- [6] SoftIntegration. *The Ch Mechanism Tool Kit – User’s Guide*. SoftIntegration, Inc, 4.5 edition, 2004. See also URL <http://www.softintegration.com>.
- [7] SoftIntegration. *The Ch Language Environment – Embedded Ch SDK User’s Guide*. SoftIntegration, Inc, 4.5 edition, 2004. See also URL <http://www.softintegration.com>.
- [8] SoftIntegration. *The Ch Language Environment – SDK User’s Guide*. SoftIntegration, Inc, 4.5 edition, 2004. See also URL <http://www.softintegration.com>.
- [9] Third Party solutions and User Contributed Code.
URL <http://www.softintegration.com/products/thirdparty/>.
- [10] Interpretive OpenGL.
URL <http://iel.ucdavis.edu/projects/opengl/>.
- [11] Kabilshkumar G Cheetancheri. Ch OpenAL package.
URL <http://chopenal.sourceforge.net/>.
- [12] Ch XML Package for Oracle C/C++ XDK.
URL <http://iel.ucdavis.edu/projects/chxml/>.
- [13] Harry H. Cheng Quingcang Yu, Bo Chen. Web-based control system design and analysis. *IEEE Control Systems Magazine*, 24(3):45–57, 2004.
- [14] SoftIntegration. *The Ch Control System Tool Kit – User’s Guide*. SoftIntegration, Inc, 2.0 edition, 2004. See also URL <http://www.softintegration.com>.

- [15] SoftIntegration. *The Ch Language Environment – CGI Toolkit User’s Guide*. SoftIntegration, Inc, 3.5.1 edition, 2004. See also URL <http://www.softintegration.com>.
- [16] Harry H. Cheng Joshua Liu. *Ch Excel – User’s Guide*. IEL, UC Davis, University of California, 2004.
- [17] Mohammed Al-Husseini Ali El-Hajj, Karim Y.Kabalan. Antenna Array Design Using Spreadsheets. *IEEE Transactions on Education*, 46(3):319–324, August 2003.
- [18] Terry E. Shoup. Using Spreadsheet Modules to Augment Tolerance Dimensioning. In *Proc. of DETC’04*, volume 2912, pages 36–44, Salt Lake City, Utah, September-October 2004. ASME.
- [19] P. W. Khong and B. L. Lim. Spreadsheet application in aircraft structural analysis. *Advances in Engineering Software*, 17(3):13–19, March 1993.
- [20] Terry E. Shoup. Using Spreadsheet Modules to Augment Design of Worm Gear. In *Proc. of DETC’03*, volume 2912, pages 36–44, Chicago, Illinois, September 2003. ASME.
- [21] Terry E. Shoup. Automated Determination of Spur Gear form factors. In *Proc. of DETC’99*, volume 2912, pages 36–44, Las Vega, Nevada, September 1999. ASME.
- [22] Didier Buchs Stanislav Chachkov. Interfacing Software Libraries from Nondeterministic Prototypes. *Design automation for embedded systems*, 8(4):327–343, 2003.
- [23] Mathias W.Richter. Iava: yet another interpreter for scripting within the Java platform. *Software - Practice and Experience*, 30(2):81–106, 2000.
- [24] Kabilshkumar G Cheetancheri. Computer aided mechanism design. URL <http://iel.ucdavis.edu/projects/chexcel>.
- [25] Harry H. Cheng. Pedagogically Effective Programming Environment for Teaching Mechanism Design. *Computer Applications in Engineering Education*, 2(1):23–39, 1994.