# ALGO
# QCM

1. **Un graphe peut être ?**
   (a) Orienté
   (b) Non orienté
   (c) A moitié orienté
   (d) Désorienté

2. **Dans un graphe orienté, le sommet x est adjacent au sommet y si ?**
   (a) Il existe un arc (x,y)
   (b) Il existe un arc (y,x)
   (c) Il existe un chemin (x,..,y)
   (d) Il existe un chemin (y,..,x)

3. **Dans un graphe orienté, un sommet de degré zéro est appelé ?**
   (a) sommet unique
   (b) sommet isolé
   (c) sommet nul
   (d) sommet perdu

4. **Un graphe orienté G défini par le triplet G=<S,A,C> est ?**
   (a) etiqueté
   (b) valué
   (c) valorisé
   (d) numéroté

5. **Dans un graphe orienté, on dit que l'arc $U = y \rightarrow x$ est ?**
   (a) incident à x vers l'extérieur
   (b) accident à x vers l'extérieur
   (c) incident à x vers l'intérieur
   (d) accident à x vers l'intérieur

6. **Dans un graphe orienté, le nombre d'arcs ayant le sommet $x$ pour extrémité terminale est appelé ?**
   (a) le demi-degré extérieur de x
   (b) le degré de x
   (c) le demi-degré intérieur de x

7. **Dans un graphe orienté, s'il existe un arc $U = y \rightarrow x$ pour tout couple de sommet $\{x, y\}$ le graphe est ?**
   (a) complet
   (b) partiel
   (c) parfait

8. **Deux arcs d'un graphe orienté sont dits adjacents si ?**
   (a) il existe deux arcs les joignant
   (b) le graphe est complet
   (c) ils ont au moins une extrémité commune

9. **L'ordre d'un graphe orienté est ?**
   (a) Le nombre d'arcs du graphe
   (b) Le nombre de sommets du graphe
   (c) Le coût du graphe
   (d) La liste triée des arcs du graphe

10. **Dans un graphe orienté valué G=<S,A,C>, les coûts sont portés par ?**
    (a) les arcs
    (b) les sommets

# QCM N°3

Lundi 16 octobre 2023

## Question 11

Considérons une suite $(u_n)$ telle que, au voisinage de $+\infty$, $u_n \sim \dfrac{(-1)^n}{n}$.

a. $\sum u_n$ converge absolument

b. $\sum u_n$ converge, mais pas absolument

c. $\sum u_n$ diverge

d. On ne peut rien dire de la nature de $\sum u_n$

## Question 12

La série $\sum \dfrac{(-1)^n}{n^2}$ :

a. converge absolument

b. converge, mais pas absolument

c. diverge

## Question 13

Soit une variable aléatoire $X$ telle que :

$$X(\Omega) = \{1, 2, 4\}, \qquad P(X{=}1) = 0.5, \qquad P(X{=}2) = 0.3 \qquad \text{et} \qquad P(X{=}4) = 0.2$$

La fonction génératrice de $X$ est définie pour tout $t \in \mathbb{R}$ par :

a. $G_X(t) = 0.5 + 0.3t + 0.2t^2$

b. $G_X(t) = 0.5t + 0.3t^2 + 0.2t^3$

c. $G_X(t) = 0.5t + 0.3t^3 + 0.2t^4$

d. Aucun des autres choix

## Question 14

Soit une variable aléatoire finie entière $X$ admettant une fonction génératrice $G_X$. On a alors :

a. $\mathrm{E}(X) = G'_X(t)$

b. $\mathrm{Var}(X) = G''_X(t) + G'_X(t) - \left(G'_X(t)\right)^2$

c. $\mathrm{Var}(X) = G''_X(1) + G'_X(1) - \left(G'_X(1)\right)^2$

d. $\mathrm{Var}(X) = G''_X(1) - G'_X(1) + \left(G'_X(1)\right)^2$

e. Aucun des autres choix

## Question 15

Soient $X$ et $Y$ deux variables aléatoires finies entières indépendantes, admettant la même fonction génératrice définie pour tout $t \in \mathbb{R}$ par

$$G_X(t) = G_Y(t) = \frac{2+t}{3}$$

Considérons la variable aléatoire $Z = X + Y$. Alors :

a. $X = Y$

b. $G_Z(t) = G_X(t) + G_Y(t)$

c. $G_Z(t) = G_X(t) \times G_Y(t)$

d. $P(Z{=}1) = \frac{4}{9}$

e. Aucun des autres choix

## Question 16

Cocher la(les) bonne(s) réponse(s)

a. $\sum \frac{1}{n}$ est une série entière

b. $\sum \frac{x}{n}$ est une série entière

c. $\sum \frac{x^n}{n}$ est une série entière

d. $\sum \frac{\sin(nx)}{n}$ est une série entière

e. Aucun des autres choix

## Question 17

Soit $(a_n)$ une suite réelle et considérons la série entière $\sum a_n x^n$. Son rayon de convergence est l'unique réel $R \in \mathbb{R}^+ \cup \{+\infty\}$ qui vérifie :

a. $\forall x \in \mathbb{R}$, $\Big(x \leqslant R \Longrightarrow \sum a_n x^n$ converge absolument$\Big)$

b. $\forall x \in \mathbb{R}$, $\Big(|x| \leqslant R \Longrightarrow \sum a_n x^n$ converge absolument$\Big)$ et $\Big(|x| > R \Longrightarrow \sum a_n x^n$ diverge$\Big)$

c. $\forall x \in \mathbb{R}$, $\Big(|x| < R \Longrightarrow \sum a_n x^n$ converge absolument$\Big)$ et $\Big(|x| \geqslant R \Longrightarrow \sum a_n x^n$ diverge$\Big)$

d. $\forall x \in \mathbb{R}$, $\Big(|x| < R \Longrightarrow \sum a_n x^n$ converge absolument$\Big)$ et $\Big(|x| > R \Longrightarrow \sum a_n x^n$ diverge$\Big)$

e. Aucun des autres choix

## Question 18

Soit une suite $(a_n)$ de termes tous non nuls telle que $\dfrac{a_{n+1}}{a_n} \xrightarrow[n\to+\infty]{} -2$.

Alors le rayon de convergence de la série entière $\sum a_n x^n$ est :

a. $R = \dfrac{1}{2}$

b. $R = -\dfrac{1}{2}$

c. $R = 2$

d. $R = -2$

## Question 19

Soit une série entière $\sum a_n x^n$ admettant un rayon de convergence $R > 0$. On note $f$ sa fonction somme, définie sur son domaine de convergence par

$$f(x) = \sum_{n=0}^{+\infty} a_n x^n$$

a. La fonction $f$ est continue sur $]-R, R[$

b. La fonction $f$ est dérivable sur $]-R, R[$ et pour tout $x \in ]-R, R[$, $\quad f'(x) = \sum_{n=1}^{+\infty} n\, a_n x^{n-1}$

c. La rayon de convergence de la série entière $\sum n\, a_n x^{n-1}$ est $R$

d. Aucun des autres choix

## Question 20

Considérons la série entière $\sum \dfrac{x^n}{n!}$. Notons $R$ son rayon de convergence et $f$ sa fonction somme, définie sur son domaine de convergence par

$$f(x) = \sum_{n=0}^{+\infty} \frac{x^n}{n!}$$

a. $R = 1$

b. $R = +\infty$

c. Pour tout $x \in ]-R, R[$, $f(x) = e^x$

d. Pour tout $x \in ]-R, R[$, $f(x) = \dfrac{1}{1-x}$

e. Aucun des autres choix

QCM 4 bis - Adjec Claus,punctu pp.289,290 fall 23

The sentences provided (21-24) are not punctuated. Choose the **one** sentence that is punctuated correctly.

21. The engineer I told you about will call you tomorrow.

A. No change needed. Correctly punctuated as is.
B. The engineer, I told you about will call you tomorrow.
C. The engineer I told you about will, call you tomorrow.
D. The engineer, I told you about, will call you tomorrow.

22. The music which I listened to was very loud.

A. The music, which I listened to was very loud.
B. The music which, I listened to was very loud.
C. The music which I listened to, was very loud.
D. No change needed. Correctly punctuated as is.

23. The Eastpak backpack which Lolita had bought on sale was already falling apart.

A. The Eastpak backpack which, Lolita had bought on sale was already falling apart.
B. The Eastpak backpack, which Lolita had bought on sale, was already falling apart.
C. The Eastpak backpack which Lolita had bought on sale, was already falling apart.
D. None of the above is correct.

24. The director Bennet Miller who has never made an action movie never works with a big budget.

A. The director Bennet Miller, who has never made an action movie never works with a big budget.
B. The director Bennet Miller, who has never made an action movie, never works with a big budget.
C. The director Bennet Miller who has never made an action movie never works with a big budget.
D. No change needed. Correctly punctuated as is.

New question: Which of the following sentence(s) is/are **RIGHT?** More than one response is possible. (25 to 30)

25.
A. The man which Stacy was arguing with was angry.
B. The man with whom Stacy was arguing with was angry.
C. The man with whom Stacy was arguing was angry.
D. The man with who Stacy was arguing was angry.

26.
A. The grant he is applying requires an essay.
B. The grant for which he is applying requires an essay.
C. The grant he is applying for requires an essay.
D. The grant of which he is applying to requires an essay.

27.

A. I donated to the fundraiser that he is supporting.
B. I donated the fundraiser that he is supporting.
C. I donated to the fundraiser who he is supporting.
D. I donated to the fundraiser he is supporting.

28.

A. I don't like to hang out with people whom drink too much.
B. I don't like to hang out with people who drink too much.
C. I don't like to hang out with people which drink too much.
D. I don't like to hang out with people drink too much.

29.

A. I must thank the president from whom I received the award.
B. I must thank the president to whom I received the award.
C. I must thank the president who I received the award from.
D. I must thank the president I received the award from.

30.

A. A man who I was talking to fell down the stairs.
B. A man to who I was talking fell down the stairs.
C. A man to whom I was talking fell down the stairs.
D. A man which I was talking to fell down the stairs.

# QCM 4 – OC S3 2023/24 (Week 16 October)

**31.** [Class article] In the 'Riding the Waves of Culture' article, which of these is **NOT** one of the four levels of cross cultural competences Trompenaars distinguishes?

a) Recognition
b) Resignation
c) Respect
d) Reconciliation

**32.** In the article which two dimensions does Trompenaars use to explain the differences between the Swedes and Americans? *Choose all that apply*

a) Individualism vs Communitarianism
b) Neutral vs Affective
c) Universalism vs Particularism
d) Specific vs Diffuse

**33.** In the article Trompenaars states that subjects like politics and religion are easily and freely discussed publicly in American culture. True or False?

a) True
b) False

**34.** In the article, what position did the Swedes have in a study about establishing social relationships?

a) Lowest ranking
b) Middle ranking
c) Highest ranking
d) None of the above – did not rank

**35.** Which of these in **NOT** one of Trompenaars 7 dimensions of culture?

a) Achievement vs Ascription
b) Sequential vs Synchronous time
c) Short vs Long term orientation
d) Internal direction vs External direction

**36.** Which of these dimensions measures a culture's tendance to separate work and professional life?

a) Individualism vs Communitarianism
b) Specific vs Diffuse
c) Neutral vs Affective
d) Universalism vs Particularism

**37.** Which of these dimensions measures a culture's tendance to follow rules?

a) Individualism vs Communitarianism
b) Specific vs Diffuse
c) Neutral vs Affective
d) Universalism vs Particularism

**38.** In the video interview, Trompenaars describes the French as _____?

a) Individualistic
b) Communitarian

**39.** In the video interview, Trompenaars states that we can look at culture like an onion with layers. The external layer being Artifacts and Behaviours and the second layer is _____?

a) Norms and Values
b) Material culture
c) Language
d) Symbols

**40.** In the video interview, Trompenaars states that the French are particularistic. True or False?

a) True
b) False

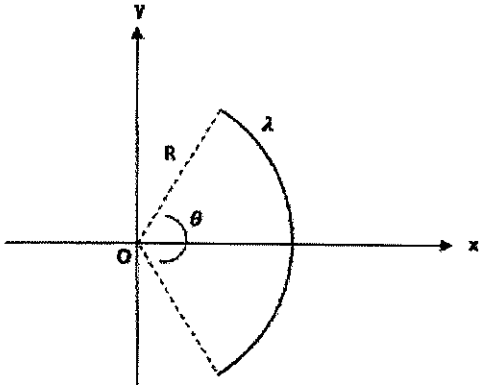
# QCM Physique – InfoS3 – 16.10

Pensez à bien lire les questions ET les réponses proposées (attention à la numérotation des réponses)

Q41. Un élément de longueur dl d'un cercle de rayon R a pour expression :

a. $dl = R.\theta$
b. $dl = \theta.dR$
c. $dl = dR.d\theta$
d. $dl = R.d\theta$

Q42. Dans la situation suivante, où l'on étudie $\overrightarrow{E(O)}$ le champ électrique créé en O par un arc de cercle de charge linéique **constante positive λ**, on peut dire que :




a. (O, $\overrightarrow{u_x}, \overrightarrow{u_y}$) est un plan de symétrie de la distribution de charges
b. (O, $\overrightarrow{u_y}, \overrightarrow{u_z}$) est un plan de symétrie de la distribution de charges
c. (O, $\overrightarrow{u_x}, \overrightarrow{u_z}$) est un plan de symétrie de la distribution de charges
d. $\overrightarrow{E(O)}$ est sur l'axe (Ox) dans le sens des x croissants

Q43. Le champ magnétique $\vec{B}$ peut être créé par :

a. Une charge fixe
b. Un ensemble de charges fixes
c. Une ou plusieurs charges en mouvement
d. Un courant électrique

Q44. L'unité d'un champ magnétique est le :

a. Volt par mètre $V.m^{-1}$
b. Volt V
c. Coulomb par mètre $C.m^{-1}$
d. Tesla T

Q45. La loi de Biot et Savart pour le champ magnétique élémentaire créé en M par un élément P de longueur dl traversé par un courant d'intensité I peut s'écrire :

a. $\overrightarrow{dB} = \frac{\mu_0}{4\pi}\frac{I.\overrightarrow{dl}\wedge\overrightarrow{PM}}{PM^3}$
b. $\overrightarrow{dB} = \frac{\mu_0}{4\pi}\frac{I.\overrightarrow{dl}\wedge\overrightarrow{PM}}{PM^2}$
c. $\overrightarrow{dB} = \frac{\mu_0}{4\pi}\frac{I.\overrightarrow{PM}\wedge\overrightarrow{dl}}{PM^3}$
d. $\overrightarrow{dB} = \frac{\mu_0}{4\pi}\frac{I.\overrightarrow{PM}\wedge\overrightarrow{dl}}{PM^2}$

Q46. La loi de Biot et Savart permet de connaître :

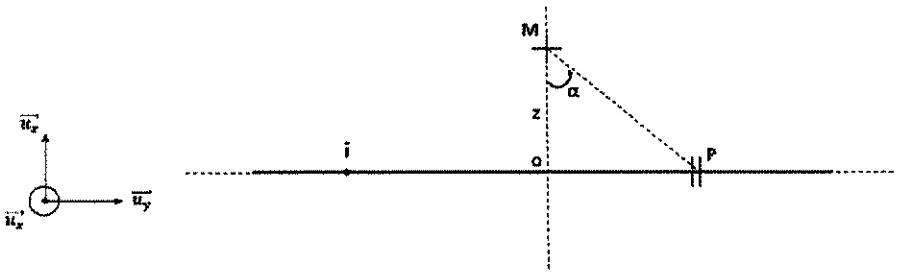
a. Les symétries d'une distribution de courants
b. La direction du champ magnétique infinitésimal créé par un élément de la distribution
c. Le sens du champ magnétique infinitésimal créé par un élément de la distribution
d. La norme du champ magnétique infinitésimal créé par un élément de la distribution

Q47. Soit une distribution de courants et un plan π passant par un point M.

a. Si π est un plan de symétrie, alors $\overrightarrow{B(M)} \in \pi$
b. Si π est un plan de symétrie, alors $\overrightarrow{B(M)} \perp \pi$
c. Si π est un plan d'antisymétrie, alors $\overrightarrow{B(M)} \in \pi$
d. Si π est un plan d'antisymétrie, alors $\overrightarrow{B(M)} \perp \pi$

Q48&49. On considère la distribution de courants suivante : un fil infini parcouru par un courant d'intensité I.

Q48. Soit z > 0. On peut dire que le champ magnétique total en M(z) vérifie :

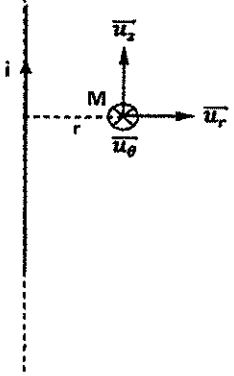



a. $\overrightarrow{B(M)}$ est selon l'axe (Oz), dans le sens des z croissants
b. $\overrightarrow{B(M)}$ est selon l'axe (Oz), dans le sens des z décroissants
c. $\overrightarrow{B(M)}$ est selon l'axe (Ox), dans le sens des x croissants
d. $\overrightarrow{B(M)}$ est selon l'axe (Ox), dans le sens des x décroissants

Q49. On désigne par $\overrightarrow{B(z)}$ le champ magnétostatique total créé au point M situé à la distance z du fil. Concernant ce champ, on peut dire :

a. $\overrightarrow{B(-z)} = -\overrightarrow{B(z)}$
b. $\overrightarrow{B(-z)} = \overrightarrow{B(z)}$
c. $\left\|\overrightarrow{B(-z)}\right\| = \left\|\overrightarrow{B(z)}\right\|$
d. $\left\|\overrightarrow{B(-z)}\right\| \neq \left\|\overrightarrow{B(z)}\right\|$

Q50. On considère un fil infini parcouru par un courant d'intensité constante I. Soit un point M situé à la distance r du fil. On désigne par $(M, \overrightarrow{u_r}, \overrightarrow{u_\theta})$ le plan passant par M et formé par les vecteurs $\overrightarrow{u_r}$ et $\overrightarrow{u_\theta}$. On peut dire que :




a. $(M, \overrightarrow{u_r}, \overrightarrow{u_\theta})$ est un plan de symétrie de la distribution de charge
b. $(M, \overrightarrow{u_r}, \overrightarrow{u_\theta})$ est un plan d'antisymétrie de la distribution de charge
c. $(M, \overrightarrow{u_r}, \overrightarrow{u_z})$ est un plan de symétrie de la distribution de charge
d. $(M, \overrightarrow{u_\theta}, \overrightarrow{u_z})$ est un plan de symétrie de la distribution de charge

# QCM 4
# Architecture des ordinateurs

**Lundi 16 octobre 2023**

**Pour toutes les questions, une ou plusieurs réponses sont possibles.**

51. Soit l'instruction suivante : `MOVE.W -(A0),D0`
    A. A0 ne change pas.
    B. A0 est décrémenté de 2.
    C. A0 est décrémenté de 4.
    D. A0 est décrémenté de 1.

52. Soit l'instruction suivante : `MOVE.L -1(A0),D0`
    A. A0 est décrémenté de 1.
    B. A0 est décrémenté de 2.
    C. A0 est décrémenté de 4.
    D. A0 ne change pas.

53. Quelles sont les deux instructions de branchements inconditionels ?
    A. JMP et JSR
    B. BSR et JSR
    C. BRA et BSR
    D. BRA et JMP

54. L'instruction BMI effectue un branchement si :
    A. V = 1
    B. N = 1
    C. None of these answers.
    D. Z = 1

55. Soient les deux instructions suivantes :

```
CMP.L D0,D1
BHI   NEXT
```

    L'instruction `BHI` effectue le branchement si :
    A. D1.L > D0.L (comparaison signée)
    B. D1.L < D0.L (comparaison signée)
    C. D1.L < D0.L (comparaison non signée)
    D. D1.L > D0.L (comparaison non signée)

56. Choisir les réponses correctes.
    A. Un nouvel élément est toujours ajouté au sommet de la pile.
    B. Un nouvel élément est toujours ajouté au bas de la pile.
    C. Un élément est toujours retiré du sommet de la pile.
    D. Un élément est toujours retiré du bas de la pile.

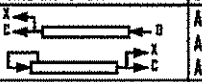
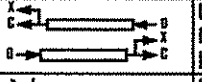
57. Le registre A7 :
    A. Pointe le sommet de la pile.
    B. Pointe le bas de la pile.
    C. Pointe le milieu de la pile.
    D. Aucune de ces réponses.

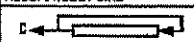
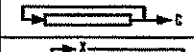
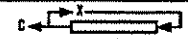
58. Choisir les réponses correctes.
    A. Un octet peut être empilé.
    B. Un mot de 16 bits peut être empilé.
    C. Un mot de 32 bits peut être empilé.
    D. Aucune de ces réponses.

59. Pour empiler une donnée :
    A. On décrémente A7 d'abord.
    B. On incrémente A7 d'abord.
    C. On ne change pas A7.
    D. Aucune de ces réponses.

60. Soit l'instruction suivante : `MOVEM.L D1-D3/A4/A5,-(A7)`
    Quelle instruction est équivalente ?
    A. `MOVEM.L D1/D3/A4-A5,-(A7)`
    B. `MOVEM.L D1/D3/A4/A5,-(A7)`
    C. `MOVEM.L A4/A5/D1/D2/D3,-(A7)`
    D. Aucune de ces réponses.

**EASy68K Quick Reference v1.8** http://www.wowgwep.com/EASy68K.htm 

| Opcode | Size | Operand | CCR | Effective Address s=source, d=destination, e=either, i=displacement | | | | | | | | | | | | Operation | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BWL | s,d | XNZVC | Dn | An | (An) | (An)+ | -(An) | (i,An) | (i,An,Rn) | abs.W | abs.L | (i,PC) | (i,PC,Rn) | #n | | |
| ABCD | B | Dy,Dx | *U*U* | e | - | - | - | - | - | - | - | - | - | - | - | $Dy_{10} + Dx_{10} + X \rightarrow Dx_{10}$ | Add BCD source and eXtend bit to |
| | | -(Ay),-(Ax) | | - | - | - | - | e | - | - | - | - | - | - | - | $-(Ay)_{10} + -(Ax)_{10} + X \rightarrow -(Ax)_{10}$ | destination, BCD result |
| ADD [4] | BWL | s,Dn | ***** | e | s | s | s | s | s | s | s | s | s | s | s[4] | s + Dn → Dn | Add binary (ADDI or ADDQ is used when |
| | | Dn,d | | e | d[4] | d | d | d | d | d | d | d | - | - | - | Dn + d → d | source is #n. Prevent ADDQ with #n.L) |
| ADDA [4] | WL | s,An | ------ | s | e | s | s | s | s | s | s | s | s | s | s | s + An → An | Add address (.W sign-extended to .L) |
| ADDI [4] | BWL | #n,d | ***** | d | - | d | d | d | d | d | d | d | - | - | s | #n + d → d | Add immediate to destination |
| ADDQ [4] | BWL | #n,d | ***** | d | d | d | d | d | d | d | d | d | - | - | s | #n + d → d | Add quick immediate (#n range: 1 to 8) |
| ADDX | BWL | Dy,Dx | ***** | e | - | - | - | - | - | - | - | - | - | - | - | Dy + Dx + X → Dx | Add source and eXtend bit to destination |
| | | -(Ay),-(Ax) | | - | - | - | - | e | - | - | - | - | - | - | - | -(Ay) + -(Ax) + X → -(Ax) | |
| AND [4] | BWL | s,Dn | -**00 | e | - | s | s | s | s | s | s | s | s | s | s[4] | s AND Dn → Dn | Logical AND source to destination |
| | | Dn,d | | e | - | d | d | d | d | d | d | d | - | - | - | Dn AND d → d | (ANDI is used when source is #n) |
| ANDI [4] | BWL | #n,d | -**00 | d | - | d | d | d | d | d | d | d | - | - | s | #n AND d → d | Logical AND immediate to destination |
| ANDI [4] | B | #n,CCR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n AND CCR → CCR | Logical AND immediate to CCR |
| ANDI [4] | W | #n,SR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n AND SR → SR | Logical AND immediate to SR (Privileged) |
| ASL | BWL | Dx,Dy | ***** | e | - | - | - | - | - | - | - | - | - | - | - |  | Arithmetic shift Dy by Dx bits left/right |
| ASR | | #n,Dy | | d | - | - | - | - | - | - | - | - | - | - | s | | Arithmetic shift Dy #n bits L/R (#n: 1 to 8) |
| | W | d | | - | - | d | d | d | d | d | d | d | - | - | - | | Arithmetic shift ds 1 bit left/right (.W only) |
| Bcc | BW[3] | address[2] | ------ | - | - | - | - | - | - | - | - | - | - | - | - | if cc true then | Branch conditionally (cc table on back) |
| | | | | | | | | | | | | | | | | address → PC | (8 or 16-bit ± offset to address) |
| BCHG | B L | Dn,d | ---*--- | e[1] | - | d | d | d | d | d | d | d | - | - | - | NOT(bit number of d) → Z | Set Z with state of specified bit in d then |
| | | #n,d | | d[1] | - | d | d | d | d | d | d | d | - | - | s | NOT(bit n of d)→ bit n of d | invert the bit in d |
| BCLR | B L | Dn,d | ---*--- | e[1] | - | d | d | d | d | d | d | d | - | - | - | NOT(bit number of d) → Z | Set Z with state of specified bit in d then |
| | | #n,d | | d[1] | - | d | d | d | d | d | d | d | - | - | s | 0 → bit number of d | clear the bit in d |
| BRA | BW[3] | address[2] | ------ | - | - | - | - | - | - | - | - | - | - | - | - | address → PC | Branch always (8 or 16-bit ± offset to addr) |
| BSET | B L | Dn,d | ---*--- | e[1] | - | d | d | d | d | d | d | d | - | - | - | NOT( bit n of d ) → Z | Set Z with state of specified bit in d then |
| | | #n,d | | d[1] | - | d | d | d | d | d | d | d | - | - | s | 1 → bit n of d | set the bit in d |
| BSR | BW[3] | address[2] | ------ | - | - | - | - | - | - | - | - | - | - | - | - | PC → -(SP); address → PC | Branch to subroutine (8 or 16-bit ± offset) |
| BTST | B L | Dn,d | ---*--- | e[1] | - | d | d | d | d | d | d | d | d | d | - | NOT( bit Dn of d ) → Z | Set Z with state of specified bit in d |
| | | #n,d | | d[1] | - | d | d | d | d | d | d | d | d | d | s | NOT(bit #n of d ) → Z | Leave the bit in d unchanged |
| CHK | W | s,Dn | -*UUU | e | - | s | s | s | s | s | s | s | s | s | s | if Dn<0 or Dn>s then TRAP | Compare Dn with 0 and upper bound [s] |
| CLR | BWL | d | -0100 | d | - | d | d | d | d | d | d | d | - | - | - | 0 → d | Clear destination to zero |
| CMP [4] | BWL | s,Dn | -**** | e | s[4] | s | s | s | s | s | s | s | s | s | s[4] | set CCR with Dn – s | Compare Dn to source |
| CMPA [4] | WL | s,An | -**** | s | e | s | s | s | s | s | s | s | s | s | s | set CCR with An – s | Compare An to source |
| CMPI [4] | BWL | #n,d | -**** | d | - | d | d | d | d | d | d | d | - | - | s | set CCR with d - #n | Compare destination to #n |
| CMPM [4] | BWL | (Ay)+,(Ax)+ | -**** | - | - | - | e | - | - | - | - | - | - | - | - | set CCR with (Ax) - (Ay) | Compare (Ax) to (Ay); Increment Ax and Ay |
| DBcc | W | Dn,address[2] | ------ | - | - | - | - | - | - | - | - | - | - | - | - | if cc false then { Dn-1 → Dn | Test condition, decrement and branch |
| | | | | | | | | | | | | | | | | if Dn <> -1 then addr →PC } | (16-bit ± offset to address) |
| DIVS | W | s,Dn | -***0 | e | - | s | s | s | s | s | s | s | s | s | s | ±32bit Dn / ±16bit s → ±Dn | Dn= [ 16-bit remainder, 16-bit quotient ] |
| DIVU | W | s,Dn | -***0 | e | - | s | s | s | s | s | s | s | s | s | s | 32bit Dn / 16bit s → Dn | Dn= [ 16-bit remainder, 16-bit quotient ] |
| EOR [4] | BWL | Dn,d | -**00 | e | - | d | d | d | d | d | d | d | - | - | s[4] | Dn XOR d → d | Logical exclusive OR Dn to destination |
| EORI [4] | BWL | #n,d | -**00 | d | - | d | d | d | d | d | d | d | - | - | s | #n XOR d → d | Logical exclusive OR #n to destination |
| EORI [4] | B | #n,CCR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n XOR CCR → CCR | Logical exclusive OR #n to CCR |
| EORI [4] | W | #n,SR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n XOR SR → SR | Logical exclusive OR #n to SR (Privileged) |
| EXG | L | Rx,Ry | ------ | e | e | - | - | - | - | - | - | - | - | - | - | register ←→ register | Exchange registers (32-bit only) |
| EXT | WL | Dn | -**00 | d | - | - | - | - | - | - | - | - | - | - | - | Dn.B → Dn.W \| Dn.W → Dn.L | Sign extend (change .B to .W or .W to .L) |
| ILLEGAL | | | ------ | - | - | - | - | - | - | - | - | - | - | - | - | PC→-(SSP); SR→-(SSP) | Generate Illegal Instruction exception |
| JMP | | d | ------ | - | - | d | - | - | d | d | d | d | d | d | - | ↑d → PC | Jump to effective address of destination |
| JSR | | d | ------ | - | - | d | - | - | d | d | d | d | d | d | - | PC → -(SP); ↑d → PC | push PC, jump to subroutine at address d |
| LEA | L | s,An | ------ | - | e | s | - | - | s | s | s | s | s | s | - | ↑s → An | Load effective address of s to An |
| LINK | | An,#n | ------ | - | - | - | - | - | - | - | - | - | - | - | - | An → -(SP); SP → An; | Create local workspace on stack |
| | | | | | | | | | | | | | | | | SP + #n → SP | (negative n to allocate space) |
| LSL | BWL | Dx,Dy | ***0* | e | - | - | - | - | - | - | - | - | - | - | - |  | Logical shift Dy, Dx bits left/right |
| LSR | | #n,Dy | | d | - | - | - | - | - | - | - | - | - | - | s | | Logical shift Dy, #n bits L/R (#n: 1 to 8) |
| | W | d | | - | - | d | d | d | d | d | d | d | - | - | - | | Logical shift d 1 bit left/right (.W only) |
| MOVE [4] | BWL | s,d | -**00 | e | s[4] | e | e | e | e | e | e | e | e | s | s[4] | s → d | Move data from source to destination |
| MOVE | W | s,CCR | ===== | s | - | s | s | s | s | s | s | s | s | s | s | s → CCR | Move source to Condition Code Register |
| MOVE | W | s,SR | ===== | s | - | s | s | s | s | s | s | s | s | s | s | s → SR | Move source to Status Register (Privileged) |
| MOVE | W | SR,d | ------ | d | - | d | d | d | d | d | d | d | - | - | - | SR → d | Move Status Register to destination |
| MOVE | L | USP,An | ------ | - | d | - | - | - | - | - | - | - | - | - | - | USP → An | Move User Stack Pointer to An (Privileged) |
| | | An,USP | | - | s | - | - | - | - | - | - | - | - | - | - | An → USP | Move An to User Stack Pointer (Privileged) |
| | BWL | s,d | XNZVC | Dn | An | (An) | (An)+ | -(An) | (i,An) | (i,An,Rn) | abs.W | abs.L | (i,PC) | (i,PC,Rn) | #n | | |

| Opcode | Size | Operand | CCR | Effective Address s=source, d=destination, e=either, i=displacement | | | | | | | | | | | | Operation | Description |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | BWL | s,d | XNZVC | Dn | An | (An) | (An)+ | -(An) | (i,An) | (i,An,Rn) | abs.W | abs.L | (i,PC) | (i,PC,Rn) | #n | | |
| MOVEA[4] | WL | s,An | ----- | s | e | s | s | s | s | s | s | s | s | s | s | s → An | Move source to An (MOVE s,An use MOVEA) |
| MOVEM[4] | WL | Rn-Rn,d | ----- | - | - | d | - | d | d | d | d | d | - | - | - | Registers → d | Move specified registers to/from memory |
| | | s,Rn-Rn | | - | - | s | s | - | s | s | s | s | s | s | - | s → Registers | (.W source is sign-extended to .L for Rn) |
| MOVEP | WL | Dn,(i,An) | ----- | s | - | - | - | - | d | - | - | - | - | - | - | Dn → (i,An)...(i+2,An)...(i+4,A. | Move Dn to/from alternate memory bytes |
| | | (i,An),Dn | | d | - | - | - | - | s | - | - | - | - | - | - | (i,An) → Dn...(i+2,An)...(i+4,A. | (Access only even or odd addresses) |
| MOVEQ[4] | L | #n,Dn | -**00 | d | - | - | - | - | - | - | - | - | - | - | s | #n → Dn | Move sign extended 8-bit #n to Dn |
| MULS | W | s,Dn | -**00 | e | - | s | s | s | s | s | s | s | s | s | s | ±16bit s * ±16bit Dn → ±Dn | Multiply signed 16-bit result signed 32-bit |
| MULU | W | s,Dn | -**00 | e | - | s | s | s | s | s | s | s | s | s | s | 16bit s * 16bit Dn → Dn | Multiply unsig'd 16-bit; result: unsig'd 32-bit |
| NBCD | B | d | *U*U* | d | - | d | d | d | d | d | d | d | - | - | - | 0 - $d_{10}$ - X → d | Negate BCD with eXtend, BCD result |
| NEG | BWL | d | ***** | d | - | d | d | d | d | d | d | d | - | - | - | 0 - d → d | Negate destination (2's complement) |
| NEGX | BWL | d | ***** | d | - | d | d | d | d | d | d | d | - | - | - | 0 - d - X → d | Negate destination with eXtend |
| NOP | | | ----- | - | - | - | - | - | - | - | - | - | - | - | - | None | No operation occurs |
| NOT | BWL | d | -**00 | d | - | d | d | d | d | d | d | d | - | - | - | NOT( d ) → d | Logical NOT destination (1's complement) |
| OR[4] | BWL | s,Dn | -**00 | e | - | s | s | s | s | s | s | s | s | s | s[4] | s OR Dn → Dn | Logical OR |
| | | Dn,d | | e | - | d | d | d | d | d | d | d | - | - | - | Dn OR d → d | (ORI is used when source is #n) |
| ORI[4] | BWL | #n,d | -**00 | d | - | d | d | d | d | d | d | d | - | - | s | #n OR d → d | Logical OR #n to destination |
| ORI[4] | B | #n,CCR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n OR CCR → CCR | Logical OR #n to CCR |
| ORI[4] | W | #n,SR | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n OR SR → SR | Logical OR #n to SR (Privileged) |
| PEA | L | s | ----- | - | - | s | - | - | s | s | s | s | s | s | - | ↑s → -(SP) | Push effective address of s onto stack |
| RESET | | | ----- | - | - | - | - | - | - | - | - | - | - | - | - | Assert RESET Line | Issue a hardware RESET (Privileged) |
| ROL | BWL | Dx,Dy | -**0* | e | - | - | - | - | - | - | - | - | - | - | - |  | Rotate Dy, Dx bits left/right (without X) |
| ROR | | #n,Dy | | d | - | - | - | - | - | - | - | - | - | - | s | | Rotate Dy, #n bits left/right (#n: 1 to 8) |
| | W | d | | - | - | d | d | d | d | d | d | d | - | - | - |  | Rotate d 1-bit left/right (.W only) |
| ROXL | BWL | Dx,Dy | ***0* | e | - | - | - | - | - | - | - | - | - | - | - |  | Rotate Dy, Dx bits L/R, X used then updated |
| ROXR | | #n,Dy | | d | - | - | - | - | - | - | - | - | - | - | s | | Rotate Dy, #n bits left/right (#n: 1 to 8) |
| | W | d | | - | - | d | d | d | d | d | d | d | - | - | - |  | Rotate destination 1-bit left/right (.W only) |
| RTE | | | ===== | - | - | - | - | - | - | - | - | - | - | - | - | (SP)+ → SR; (SP)+ → PC | Return from exception (Privileged) |
| RTR | | | ===== | - | - | - | - | - | - | - | - | - | - | - | - | (SP)+ → CCR, (SP)+ → PC | Return from subroutine and restore CCR |
| RTS | | | ----- | - | - | - | - | - | - | - | - | - | - | - | - | (SP)+ → PC | Return from subroutine |
| SBCD | B | Dy,Dx | *U*U* | e | - | - | - | - | - | - | - | - | - | - | - | $Dx_{B}$ - $Dy_{B}$ - X → $Dx_{B}$ | Subtract BCD source and eXtend bit from |
| | | -(Ay),-(Ax) | | - | - | - | - | e | - | - | - | - | - | - | - | $-(Ax)_{B}$ - $-(Ay)_{B}$ - X → $-(Ax)_{B}$ | destination, BCD result |
| Scc | B | d | ----- | d | - | d | d | d | d | d | d | d | - | - | - | If cc is true then 1's → d | If cc true then d.B = 11111111 |
| | | | | | | | | | | | | | | | | else 0's → d | else d.B = 00000000 |
| STOP | | #n | ===== | - | - | - | - | - | - | - | - | - | - | - | s | #n → SR; STOP | Move #n to SR, stop processor (Privileged) |
| SUB[4] | BWL | s,Dn | ***** | e | s | s | s | s | s | s | s | s | s | s | s[4] | Dn - s → Dn | Subtract binary (SUBI or SUBQ used when |
| | | Dn,d | | e | d[4] | d | d | d | d | d | d | d | - | - | - | d - Dn → d | source is #n. Prevent SUBQ with #n.L) |
| SUBA[4] | WL | s,An | ----- | s | e | s | s | s | s | s | s | s | s | s | s | An - s → An | Subtract address (.W sign-extended to .L) |
| SUBI[4] | BWL | #n,d | ***** | d | - | d | d | d | d | d | d | d | - | - | s | d - #n → d | Subtract immediate from destination |
| SUBQ[4] | BWL | #n,d | ***** | d | d | d | d | d | d | d | d | d | - | - | s | d - #n → d | Subtract quick immediate (#n range: 1 to 8) |
| SUBX | BWL | Dy,Dx | ***** | e | - | - | - | - | - | - | - | - | - | - | - | Dx - Dy - X → Dx | Subtract source and eXtend bit from |
| | | -(Ay),-(Ax) | | - | - | - | - | e | - | - | - | - | - | - | - | -(Ax) - -(Ay) - X → -(Ax) | destination |
| SWAP | W | Dn | -**00 | d | - | - | - | - | - | - | - | - | - | - | - | bits[31:16]←→bits[15:0] | Exchange the 16-bit halves of Dn |
| TAS | B | d | -**00 | d | - | d | d | d | d | d | d | d | - | - | - | test d→CCR; 1 →bit7 of d | N and Z set to reflect d, bit7 of d set to 1 |
| TRAP | | #n | ----- | - | - | - | - | - | - | - | - | - | - | - | s | PC→-(SSP),SR→-(SSP); | Push PC and SR, PC set by vector table #n |
| | | | | | | | | | | | | | | | | (vector table entry) → PC | (#n range: 0 to 15) |
| TRAPV | | | ----- | - | - | - | - | - | - | - | - | - | - | - | - | If V then TRAP #7 | If overflow, execute an Overflow TRAP |
| TST | BWL | d | -**00 | d | - | d | d | d | d | d | d | d | - | - | - | test d → CCR | N and Z set to reflect destination |
| UNLK | | An | ----- | - | d | - | - | - | - | - | - | - | - | - | - | An → SP; (SP)+ → An | Remove local workspace from stack |
| | BWL | s,d | XNZVC | Dn | An | (An) | (An)+ | -(An) | (i,An) | (i,An,Rn) | abs.W | abs.L | (i,PC) | (i,PC,Rn) | #n | | |

| Condition Tests (+ OR, ! NOT, ⊕ XOR; * Unsigned, ° Alternate cc) | | | | | |
|---|---|---|---|---|---|
| cc | Condition | Test | cc | Condition | Test |
| T | true | 1 | VC | overflow clear | !V |
| F | false | 0 | VS | overflow set | V |
| HI* | higher than | !(C + Z) | PL | plus | !N |
| LS* | lower or same | C + Z | MI | minus | N |
| HS*, CC° | higher or same | !C | GE | greater or equal | !(N ⊕ V) |
| LO*, CS° | lower than | C | LT | less than | (N ⊕ V) |
| NE | not equal | !Z | GT | greater than | !((N ⊕ V) + Z) |
| EQ | equal | Z | LE | less or equal | (N ⊕ V) + Z |

An Address register (16/32-bit, n=0-7)
Dn Data register (8/16/32-bit, n=0-7)
Rn any data or address register
s Source, d Destination
e Either source or destination
#n Immediate data, i Displacement
BCD Binary Coded Decimal
↑ Effective address
1 Long only; all others are byte only
2 Assembler calculates offset
3 Branch sizes: .B or .S -128 to +127 bytes, .W or .L -32768 to +32767 bytes
4 Assembler automatically uses A, I, Q or M form if possible. Use #n.L to prevent Quick optimization

SSP Supervisor Stack Pointer (32-bit)
USP User Stack Pointer (32-bit)
SP Active Stack Pointer (same as A7)
PC Program Counter (24-bit)
SR Status Register (16-bit)
CCR Condition Code Register (lower 8-bits of SR)
N negative, Z zero, V overflow, C carry, X extend
* set according to operation's result, = set directly
- not effected, 0 cleared, 1 set, U undefined

Revised by Peter Csaszar, Lawrence Tech University – 2004-2006